SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

Marija Vlajčić

PREGLED I ANALIZA PERFORMANSI SOFTVERSKI DEFINIRANIH MREŽA

DIPLOMSKI RAD

Zagreb, 2018.

Sveučilište u Zagrebu

Fakultet prometnih znanosti

DIPLOMSKI RAD

PREGLED I ANALIZA PERFORMANSI SOFTVERSKI DEFINIRANIH MREŽA

REVIEW AND PERFORMANCE ANALYSIS OF SOFTWARE DEFINED NETWORKS

Mentor: Doc. dr. sc. Ivan Grgurević, dipl. ing.

Student: Marija Vlajčić, 0135222178

Datum obrane: rujan 2018.

Zagreb, rujan 2018.

PREGLED I ANALIZA PERFORMANSI SOFTVERSKI DEFINIRANIH MREŽA

SAŽETAK

U današnje vrijeme, kada se traži odgovor na potrebu standardizacije tehnologija i terminali predstavljaju logičke uređaje, softverski definirano umrežavanje nudi inovativna rješenja. Tradicionalni načini umrežavanja dali su glavnu ulogu hardveru, odnosno mrežnim uređajima, što dovodi do određenih problema prilikom daljnjeg održavanja mreže. S druge strane, suvremeni mrežni softver je izrazito složene i zahtjevne strukture te je potrebno detaljnije objasniti procese metoda i tehnika strukture mrežnog softvera. U diplomskom radu opisane su osnovne značajke softverski definiranih mreža (SDN) i pripadajuća arhitektura mreže sa svojim dijelovima. Prikazane su funkcije, tablica usmjeravanja i poruke standardiziranog OpenFlow protokola. Kroz teorijski prikaz navedeno je što je bitno za planiranje SDN mreža, a zatim je prikazano konfiguracijom i simulacijom u programskom alatu GNS3. Za kraj su analizirane performanse SDN mreža u odnosu na dosadašnje tradicionalne mreže.

KLJUČNE RIJEČI: SDN mreža, arhitektura mreže, OpenFlow protokol, planiranje mreže, konfiguracija, NFV, SD-WAN.

SUMMARY

Nowadays when the need for standardization of technology is sought and terminals are logical devices, software-defined networking offers innovative solutions. Traditional networking methods have played a major role in hardware or network devices, leading to certain problems when maintaining network further. On the other hand, modern networking software is a highly complex and demanding structure and needs to be explained more in detail in the methods and techniques of the network software structure. The graduate thesis describes the basic features of software-defined networks (SDN) and the related network architecture with their parts. The features, routing tables, and messages of the standard OpenFlow protocol are displayed. The theory states what is important to the planning of the SDN network, and then presented by configuration and simulation in the GNS3 programming tool. For the end, the performance of the SDN network was analyzed in relation to the traditional networks.

KEYWORDS: SDN network, network architecture, OpenFlow protocol, network planning, configuration, NFV, SD-WAN.

Sadržaj

1. Uvod 1
2. Osnovne značajke SDN mreža 3
3. Arhitektura SDN mreža 7
4. Pregled Open Flow protokola 12
   4.1. Tablica usmjeravanja Open Flow protokola 15
      4.1.1. Podudaranja tablica usmjeravanja 16
      4.1.2. Cjevovod tablica usmjeravanja 18
   4.2. Poruke Open Flow protokola 19
   4.3. Zaglavlje OpenFlow protokola 22
5. Planiranje SDN mreža 25
6. Prikaz konfiguracije i simulacije SDN mreža u programskom alatu GNS3 32
   6.1 Simulacija Mininet SDN platforme 34
   6.2 Mjerenje osnovnih mrežnih parametara SDN mreže 37
7. Analiza performansi SDN mreža 42
   7.1. Područja primjene SDN mreža 43
      7.1.1. NFV koncept 45
      7.1.2. SD-WAN 47
   7.2. Poslovne prednosti SDN mreža 49
8. Zaključak 52

Literatura 54

Popis kratica i akronima 58

Popis slika i tablica 59

1. Uvod

U početku razvoja računalnih mreža, pojavile su se zatvorene, vlasničke mreže. Kao takve, bile su strogo definirane pravilima samo određenog proizvođača opreme, koristila se zasebna tehnologija i nije bila moguća prilagodba zahtjevima korisnika. Točnije, oprema proizvođača, mogla je funkcionirati samo u mreži za koju je proizvedena, što nije bio problem sve dok konkurencija nije počela odgovarati na potrebe korisnika. U novije vrijeme se računalne mreže baziraju na standardiziranoj tehnologiji, a mrežna oprema uključuje logičku domenu. Drugim riječima, u ranijim strategijama razvoja računalnih mreža glavnu ulogu imao je hardver, a danas se teži standardizaciji mrežnih procesa i glavnu ulogu u mrežnom upravljanju preuzima softver. Softverski definirano umrežavanje predstavlja koncept optimizacije mrežnih resursa odvajanjem upravljačkog dijela i podataka te samim time postupke održavanja i nadogradnje mreže čini jednostavnijim.

U diplomskom radu dan je pregled i analiza performansi softverski definiranih mreža (SDN) u odnosu na dosadašnja konvencionalna rješenja. SDN je dinamično i isplativo rješenje, idealno za današnje aplikacije velike propusnosti. Pregled performansi dan je u prikazu dijela konfiguracije i simulacije SDN mreže u programskom alatu GNS3.

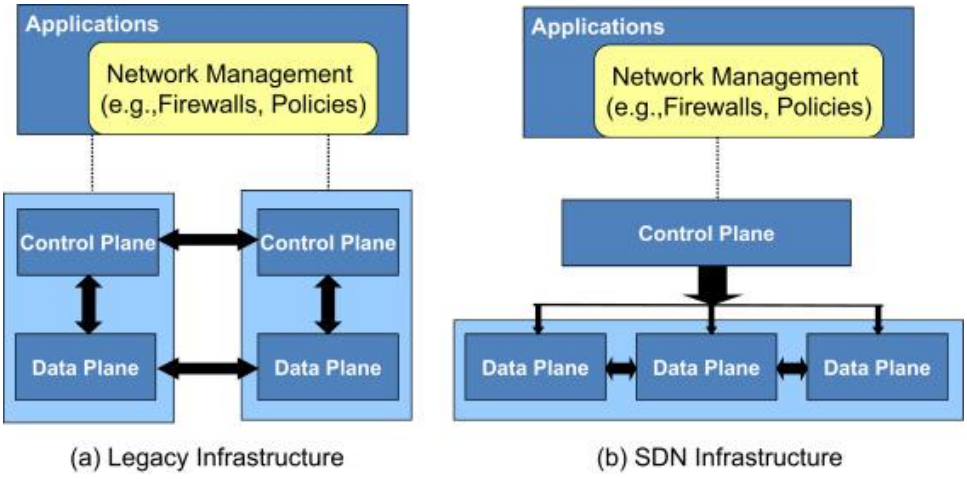
Diplomski rad sastoji se od osam (8) poglavlja:

1. Uvod
2. Osnovne značajke SDN mreža
3. Arhitektura SDN mreža
4. Pregled Open Flow protokola
5. Planiranje SDN mreža
6. Prikaz konfiguracije i simulacije SDN mreže u programskom alatu GNS3
7. Analiza performansi SDN mreža
8. Zaključak.

Za početak će biti navedene osnovne značajke softverski definiranih mreža, funkcije pojedinih dijelova i princip rada. Zatim će se prikazati arhitektura mreže i način na koji se odvija komunikacija između mrežnih uređaja. U sljedećem poglavlju detaljno će se definirati karakteristike *OpenFlow* protokola i način popunjavanja tablice usmjeravanja. Zatim će biti spomenuto planiranje mreže kod softverski definiranih mreža te će se provesti istraživanje performansi softverski definiranih mreža, poboljšanja koja donose i mogućnosti primjene. Kao takav, cilj diplomskog rada je napraviti pregled i analizu performansi softverski definiranih mreža, što će biti vidljivo i prikazom rezultata mjerenja performansi provedenog u programskom alatu GNS3.

2. Osnovne značajke SDN mreža

Softverski definirane mreže (engl. *Software Defined Networking*, SDN) predstavljaju oblik mrežne arhitekture koji nastoji optimizirati mrežne resurse i prilagoditi mrežu promjenama poslovnih potreba, aplikacija i prometa. Djeluje odvajanjem upravljačkog dijela mreže i podataka, stvarajući programibilnu infrastrukturu koja se razlikuje od fizičkih uređaja i dosadašnjih konvencionalnih mreža (Slika 1.). U SDN-u funkcije mrežne arhitekture, upravljanja, analitike i automatizacije postaju posao SDN kontrolera. Budući da SDN kontroleri nisu mrežni uređaji, mogu iskoristiti prednosti mjerila, pohrane podataka i dostupnosti suvremenih resursa računalstva u oblaku. U novije vrijeme SDN kontroleri izgrađuju se na otvorenim platformama, koristeći otvorene standarde, što im omogućuje korištenje i upravljanje mrežne opreme različitih proizvođača. SDN pruža široku lepezu poslovnih prednosti. Razdvajanje upravljačkih i prijenosnih slojeva povećava fleksibilnost i ubrzava vrijeme plasiranja novih aplikacija na tržište. Sposobnost brže reakcije na probleme i prekide poboljšava dostupnost mreže, a programabilnost olakšava IT organizacijama automatizaciju mrežnih funkcija, smanjujući operativne troškove. SDN je dinamična, isplativa i prilagodljiva mrežna arhitektura u razvoju što ju čini idealnom za dinamičnu prirodu velike propusnosti današnjih aplikacija. [1]

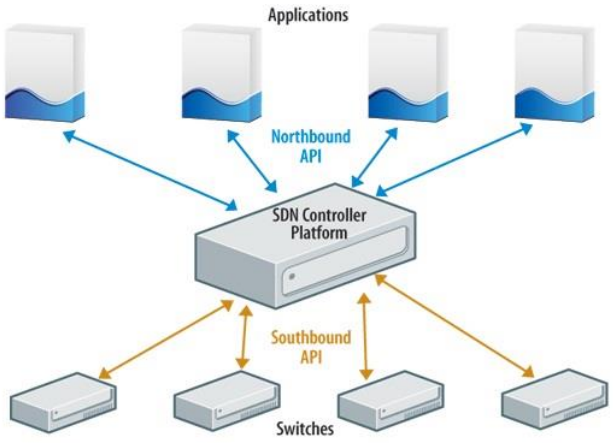


Slika 1. Razlika između konvencionalne i SDN infrastrukture

Izvor [24]

Cilj SDN-a je omogućiti računalstvu u oblaku, mrežnim inženjerima i administratorima brzu reakciju na promjenu poslovnih zahtjeva putem centralizirane upravljačke jedinice. SDN obuhvaća više vrsta mrežnih tehnologija osmišljenih kako bi mreža bila fleksibilnija te kako bi podržala virtualiziranu infrastrukturu poslužitelja i zadovoljila potrebe velike propusnosti današnjih aplikacija. Softverski definirano umrežavanje izvorno je definiralo pristup projektiranju, izgradnji i upravljanju mrežama na način da se razdvoji mrežno upravljanje i prosljeđivanje. Time je omogućeno da mrežno upravljanje postane izravno programibilno i da se temeljna infrastruktura proširi za aplikacije velike propusnosti i mrežne usluge kao što su SDN računalstvo u oblaku i mobilne mreže. [3]

Softverski definirani davatelji mrežnih usluga nude širok izbor konkurentskih arhitektura, ali najzastupljenija je metoda umrežavanja putem središnje upravljačke jedinice, razdvajanjem upravljačke logike na računalne resurse izvan uređaja. Sva softverska rješenja definirana mrežnim rješenjima imaju neku verziju SDN kontrolera, kao i programskih sučelja, *Southbound* API[1] (engl. *Application Programming Interface*) i *Northbound* API (Slika 2.). SDN kontroleri nude centralizirani prikaz cjelokupne mreže i omogućuju mrežnim administratorima upravljanje osnovnim sustavima. *Southbound* API-ji koriste se za prijenos informacija na *router*-e[2] i *switch*-eve[3]. *OpenFlow* se smatra prvim standardom u SDN-u, a izvorno je bio *Southbound* API i ostaje jedan od najčešće korištenih protokola. *Northbound* API-ji koriste se za komunikaciju s aplikacijama i pomažu mrežnim administratorima da programski oblikuju promet i implementiraju usluge. [4]



[1] API: engl. *Application Programming Interface* ili sučelje za programiranje aplikacija je skup programskih definicija, protokola i alata za izgradnju aplikacijskog softvera.

[2] *Router*: usmjerivač ili uređaj koji usmjerava podatkovne pakete na njihovom putu kroz mrežu.

[3] Switch: preklopnik ili uređaj koji omogućava spajanje dva ili više računala u jednu mrežu.

Slika 2. Logički prikaz SDN mreže

Izvor [5]

Često se u raznoj literaturi *OpenFlow* spominje kao sinonim za softverski definirano umrežavanje, ali to je samo jedan element u cjelokupnoj SDN arhitekturi. *OpenFlow* je otvoreni standard za komunikacijski protokol koji omogućuje povezivanje upravljačke razine s razinom prosljeđivanja. Važno je napomenuti da *OpenFlow* nije jedini dostupan razvojni protokol za SDN.

Društveni mediji, mobilni uređaji i računalstvo u oblaku dovode tradicionalne mreže do njihovih granica u razvoju. Pohrana informacija i računalstvo su profitirali inovacijama u virtualizaciji i automatizaciji, no prisutna su ograničenja u mreži. Mreža se često održava ručnim mrežnim operacijama i ponavljanjem postupaka na svakom mrežnom uređaju zasebno. Softverski definirano umrežavanje pruža mogućnost unaprjeđenja dosadašnjih podatkovnih centara , čineći pritom način upravljanja mrežom mnogo fleksibilnijim.

Dosadašnja istraživanja pokazala su brojne prednosti implementacije SDN-a u mnogim knjigama poznatih autora, što će kasnije kroz poglavlja biti i objašnjeno. [1], [6], [7], [4]

Važne performanse u vidu bržeg otklanjanja pogreške, lakšeg održavanja i upravljanja mrežom, uvidjeli su i razni proizvođači opreme tzv. vendori, što je rezultiralo plasmanom raznih rješenja na tržište. Proizvođač mrežne opreme *Juniper* tako je posljednje predstavio SDN povezan s drugom tehnologijom, zvanom virtualizacija mrežnih funkcija (engl. *Network function virtualization*, NFV). NFV nudi mogućnost virtualizacije mrežnih funkcija temeljenih na uređajima, kao što su vatrozidi, balanseri opterećenja i akceleratori za WAN (engl. *Wide Area Network*). Centralizirano upravljanje koje pruža SDN može učinkovito pratiti virtualne funkcije mreže koje omogućava NFV. [8]

Također proizvođači opreme *Cisco Systems* i *Microsoft* u znanstvenim člancima opisuju svoje inačice kao fleksibilno i programibilno rješenje s platformom sposobnom za rješavanje najzahtjevnijih mrežnih potreba. [9], [10]

Američka zaklada pod nazivom *Open networking fondation* (skraćeno ONF), okuplja neke od vodećih telekomunikacijskih operatora, vendora i sistem integratora u svrhu iskorištavanja SDN principa, koristeći otvorene platforme i definirane standarde za izgradnju mreža. [11]

Za simulacije mreža i mrežnih protokola u istraživačke svrhe, vrlo često se koristi programski alat GNS3, koji će biti korišten i u ovom radu. [12], [13]

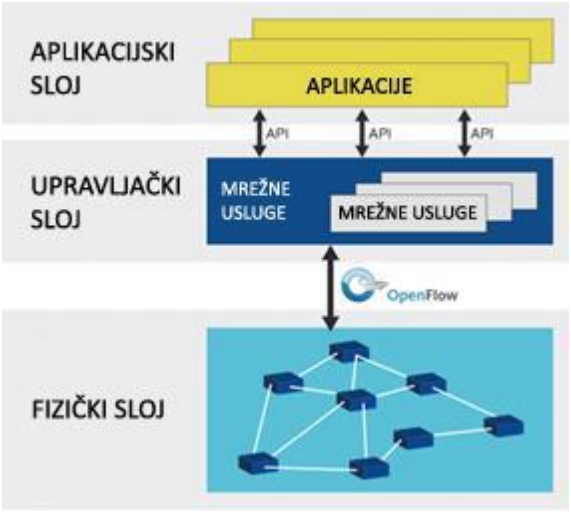
U slučaju da je potrebna prilagodba ili promjena na nekoj od postojećih tehnologija koristi se kompleksniji alat *Riverbed Modeler*, pa se razni autori bave takvim pitanjima. [14]

Primjerice proizvođač opreme *Cisco*, izdao je knjigu s detaljnim opisom kako i prema kojim načelima se može konfigurirati SDN mreža koristeći programski alat GNS3. [15]

Pregledom dosadašnjih dostupnih istraživanja prepoznati su i znanstveni članci koji obrađuju problematiku različitih karakteristika SDN mreža koristeći programski alat GNS3. [16], [17], [18]

3. Arhitektura SDN mreža

U doba skokovitog razvoja novih mrežnih aplikacija velike propusnosti te povećane potrebe za podatkovnim prometom, sve je veća odgovornost na brzini obrade podataka i samom upravljanju paketima u mrežnim uređajima poput usmjeritelja, komutatora, vatrozida i sl. Upravo ideja razdvajanja vertikalne mrežne infrastrukture na zaseban upravljački i podatkovni sloj te otvaranje mogućnosti progamabilnosti mreža, prikazuje srž koncepta softverskog upravljanja mrežom. U sustavu dizajniranom na taj način, ključnu ulogu ima središnja upravljačka jedinica, odgovorna za upravljanje uređajima pod svojom domenom, koji u tom slučaju tvore fizičku mrežnu infrastrukturu.



Slika 3. Arhitektura SDN mreže

Izvor [19]

Kao što je vidljivo sa Slike 3., arhitektura SDN mreže sastoji se od tri sloja. Središnji sloj, odnosno upravljački obavlja sve složene funkcije, uključujući usmjeravanje, određivanje pravila i sigurnosne provjere. Ova razina objedinjuje upravljanje podatkovnim prometom, a sastoji se od jednog ili više SDN poslužitelja. SDN upravljačka jedinica, u daljnjem tekstu kontroler, definira tok podataka koji se pojavljuju u SDN podatkovnoj razini ili fizičkom sloju. Svaki tijek podataka[4] prvo mora

[4] Tijek ili tok podataka: niz paketa koji prolaze kroz mrežu i dijele neka zajednička obilježja. Npr, tok može biti sastavljen od svih paketa s istim izvornim i odredišnim IP adresama, ili s paketima s istim VLAN identifikatorom.

dobiti dozvolu kontrolera, što potvrđuje da je komunikacija mrežnim resursima dopuštena. Ako je kontroler dopustio određeni podatkovni tijek, izračunava rutu i javlja informaciju o propuštanju tog tijeka svakom od *switch*-eva duž rute. Uz sve složene funkcije koje obuhvaća kontroler, isključivo on popunjava tablice usmjeravanja, a *switch*-evi jednostavno dalje upravljaju usmjeravanjem. Za komunikaciju između kontrolera i *switch*-eva koristi se standardizirani *OpenFlow* protokol i API-ji.

U SDN arhitekturi kontroler obavlja sljedeće funkcije:

- Definiranje korisničke opreme kao što su prijenosna računala, stolna računala, printeri, mobilni uređaji itd.
- Definiranje mrežnih uređaja koji upotpunjuju infrastrukturu mreže, kao što su switchevi, router-i i bežične pristupne točke.
- Upravljanje topologijom mreže, održavanjem informacija o detaljima povezanosti između mrežnih uređaja i korisničke opreme na koju su izravno povezani.
- Upravljanje prometnim tokom, održavanjem tablica usmjeravanja toka kojima upravlja kontroler i obavljanje svih potrebnih koordinacija s uređajima kako bi se osigurala sinkronizacija ulaznog toka uređaja s bazom podataka tablica usmjeravanja. [20]

U SDN arhitekturi *switch*-evi obavljaju sljedeće funkcije:

- Enkapsulira[5] i prosljeđuje prvi paket prijenosa na SDN kontroler, omogućujući kontroleru odluku treba li se tijek podataka dodati u tablicu usmjeravanja *switch*-a.
- Prosljeđuje dolazne pakete s odgovarajućeg porta na temelju tablice usmjeravanja. Tablica usmjeravanja može sadržavati informacije o prioritetima posluživanja koje određuje kontroler.
- Može privremeno ili trajno ispustiti pakete na određenoj ruti ukoliko to odredi kontroler. Ispuštanje paketa može se iskoristiti u sigurnosne svrhe, sprječavajući napade uskraćivanja usluga (engl. *Denial of Service*, DoS) ili zahtjeve za upravljanje prometom.

---

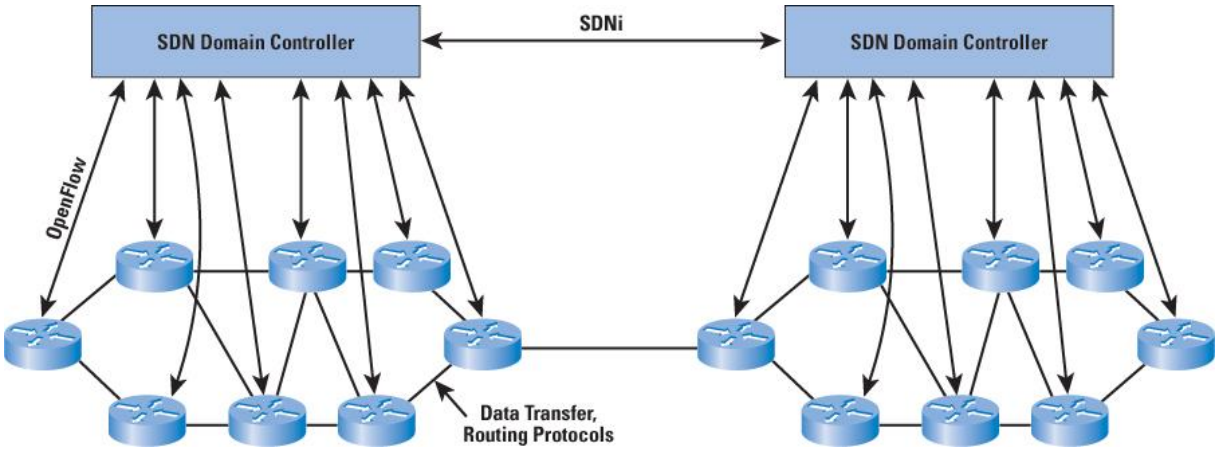
[5] Enkapsulacija: postupak pakiranja podataka, od 7. sloja prema 1. Sloju OSI referentnog modela, u oblik pogodan za prijenos komunikacijskim vezama.

Arhitekturu SDN-a može se promatrati i preko tri apstrakcijska sloja: prosljeđivanja, distribucije i specifikacije. Apstrakcijski sloj prosljeđivanja, potpuno neovisan o fizičkim svojstvima mreže, obavlja ulogu izvršavanja i podrške aplikacijski proslijeđenim zahtjevima. Sloj distribucije, uz logički centraliziranu ulogu upravljanja mrežnim uređajima te prikupljanja informacija o njihovom radu i međusobnoj povezanosti, stvara temelj za rad mrežnih aplikacija. Način na koji mrežne aplikacije realiziraju svoje primarne funkcije, bez zadiranja u samu implementaciju na fizičkoj razini, pripada apstrakcijskom sloju specifikacije koji ovakve zahtjeve rješava upotrebom virtualizacije te programskih jezika.

Ako pogledamo arhitekturu SDN mreže s aspekta velikih poduzeća , u velikoj mreži je vrlo teško, gotovo nemoguće i nepoželjno, postaviti jednog kontrolera za upravljanje svim mrežnim uređajima. U takvim situacijama koriste se zasebne SDN domene (Slika 4.).

Razlozi za korištenje SDN domena uključuju sljedeće:

- Skalabilnost: broj uređaja kojima SDN kontroler može uspješno upravljati ograničen je. Dakle, ukoliko se radi o velikoj mreži, trebat će implementirati više SDN kontrolera.
- Privatnost: moguće je odabrati primjenu različitih pravila o privatnosti na različitim domenama SDN-a. Na primjer, SDN domena može zatražiti da neke informacije o mreži u ovoj domeni, npr. topologija mreže ne budu otkrivene entitetima izvan mreže.
- Dijeljena implementacija: transportna mreža može se sastojati od dijelova tradicionalne i novije infrastrukture. Dijeljenje mreže u više pojedinačno upravljanih SDN domena omogućuje veću fleksibilnost. [21]



Slika 4. Prikaz SDN domena

Izvor [22]

Postojanje višestrukih domena stvara potrebu za komunikacijom između pojedinačnih kontrolera, što bi moglo biti omogućeno standardiziranim protokolom za razmjenu informacija usmjeravanja. IETF (engl. *Internet Engineering Task Force*) trenutno radi na razvoju protokola, nazvanog SDNi (engl. *Software Defined Networking interface*), predviđenog za povezivanje kontrolera domena SDN-a.

SDNi funkcije uključuju:

- Postavljanje koordinata protokola, što potječe od aplikacija kojima je poznat zahtjev za rutom, QoS (engl. *Quality of Service*) i dogovorena pravila na razini usluge za različite SDN domene.
- Informacije o dostupnosti razmjene podataka između domena, kako bi se pojednostavilo usmjeravanje. Ova razmjena informacija omogućit će da jedan tijek podataka prelazi više SDN-ova i da svaki kontroler odabere najpogodniju rutu.

Vrste poruka za SDNi uključuju sljedeće:

- Ažuriranje dostupnosti
- Zahtjev za postavljanje / uklanjanje / ažuriranje tijeka podataka (uključujući zahtjeve kao što su QoS, brzina prijenosa podataka, latencija itd.)
- Ažuriranje mogućnosti (uključujući značajke kao što su brzina prijenosa podataka i QoS te mogućnosti sustava i softvera dostupne unutar domene)

Riječ je o vrlo fleksibilnoj arhitekturi, koja može raditi s različitim tipovima *switch*-eva i različitim protokolskim slojevima. SDN kontroleri i *switch*-evi mogu se provesti

za *Ethernet*[6] *switch*-eve (podatkovni sloj), internet *router*-e (mrežni sloj), prijenosno usmjeravanje (transportni sloj) ili usmjeravanje aplikacijskog sloja. SDN se oslanja na uobičajene funkcije koje se nalaze na mrežnim uređajima, što u osnovi uključuje prosljeđivanje paketa na temelju nekog oblika definicije prometnog tijeka. Jednostavnije rečeno, SDN kontroler upravlja usmjeravanjem *switch*-eva u SDN. To se upravljanje vrši pomoću dobavljačkog neutralnog API-ja koji kontroleru omogućuje obradu velikog broja zahtjeva operatora bez promjene obilježja nižih slojeva mreže, uključujući topologiju. Odvajanjem upravljačkih i podatkovnih slojeva, aplikacijama je omogućena koncentracija na jedan, konkretni mrežni uređaj bez potrebe o poznavanju pojedinosti načina rada uređaja. Mrežnim aplikacijama upravljaju jedinstveni API-ji. Tako je moguće brzo kreirati i implementirati nove aplikacije visoke propusnosti kako bi se zadovoljili današnji specifični zahtjevi za performansama i sigurnosti. [23]

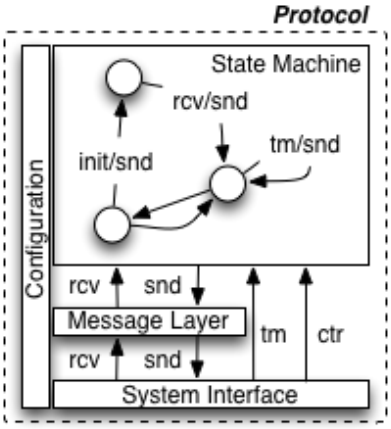
[6] *Ethernet*: najčešće korištena tehnologija umrežavanja za lokalne mreže.

4. Pregled Open Flow protokola

Kako bi se koncept SDN mreže zaista primijenio u praksi, potrebno je ispuniti dva uvjeta:

1. Zajednička logička arhitektura na svim mrežnim uređajima kojima upravlja SDN kontroler. Logičku arhitekturu moguće je implementirati na različite načine od različitih dobavljača opreme i s različitim vrstama mrežnih uređaja, dokle god je SDN kontroleru u potpunosti omogućeno obavljanje svih njegovih funkcija.
2. Standardizirani protokol za komunikaciju između SDN kontrolera i mrežnih uređaja.

Oba uvjeta ispunjava *OpenFlow*, koji je i protokol između SDN kontrolera i mrežnih uređaja, kao i specifikacija logičke arhitekture mreže. *OpenFlow* je definiran u dokumentu *OpenFlow Switch Specification*, objavljenoj od *Open Networking Foundation* (kratica ONF). ONF je Američka zaklada koja okuplja neke od vodećih telekomunikacijskih operatora, vendora i sistem integratora u svrhu iskorištavanja SDN principa, koristeći otvorene platforme i definirane standarde za izgradnju mreža.



Slika 5. Prikaz komponenti OpenFlow protokola

Izvor [24]

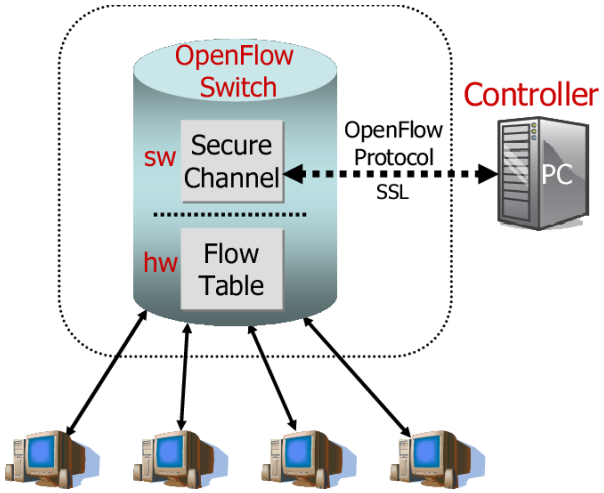
OpenFlow protokol može se razvrstati u četiri komponente: *Message Layer, State Machine, System Interface, Configuration i Data Model*. Slikom 5. prikazana je interakcija između komponenti, a Tablicom 1. opisana je funkcija svake pojedine komponente. [6]

Tablica 1. Funkcija komponenti OpenFlow protokola

Komponenta	Opis
Message Layer	*Message Layer* je temelj protokola. Definira ispravnu strukturu i semantiku poruka te podržava mogućnost konstruiranja, kopiranja, usporedbe, ispisa i rukovanja porukama.
State Machine	*State Machine* definira ponašanje protokola na nižoj razini. Koristi se za opisivanje radnji kao što su: pregovaranje, otkrivanje sposobnosti, kontrola prometnog toka, isporuka itd.
System Interface	*System Interface* definira kako protokol komunicira s okolinom. Spaja obavezna i moguća sučelja zajedničkom namjenom, kao što su npr. TLS i TCP transportni protokoli.
Configuration	Gotovo svi aspekti protokola imaju konfiguracije ili početne vrijednosti. Konfiguracija može obuhvatiti onoliko podataka kolika je zadana veličina međuspremnika i odgovara na certifikate X.509.
Data Model	Iz drugog kuta gledanja,. bitno je razumjeti *Data Model* OpenFlow protokola. Svaki *switch* održava relacijski model podataka koji sadrži atribute za svaku OpenFlow apstrakciju. Ti atributi mogu opisivati sposobnost apstrakcije, stanje konfiguracije ili neki skup aktualnih statistika.

Izvor [24]

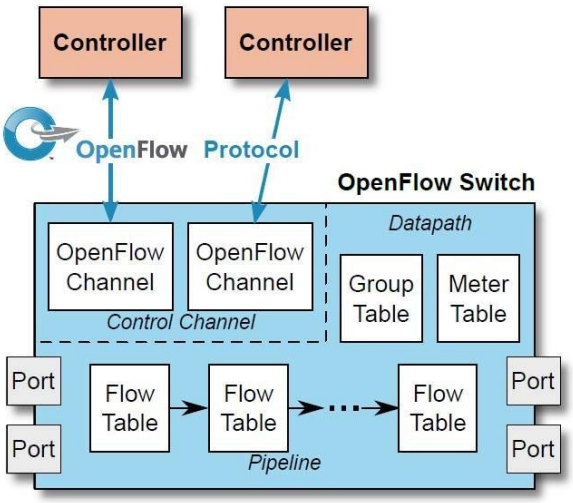
Slikom 6. prikazana je osnovna arhitektura *OpenFlow* protokola. Ukratko, SDN kontroler komunicira s *OpenFlow* kompatibilnim preklopnicima koristeći *OpenFlow* protokol koji pokreće SSL[7] (engl. *Secure Sockets Layer*). Svi *switch*-evi su povezani međusobno ili s krajnjim korisnicima koji čine izvorišta i odredišta tokova paketa.



Slika 6. Prikaz arhitekture OpenFlow protokola

Izvor [25]

Unutar svakog *switch*-a implementiran je niz tablica usmjeravanja koje se koriste za upravljanje tokovima paketa (Slika 7.). [23]



Slika 7. Prikaz OpenFlow switch-a

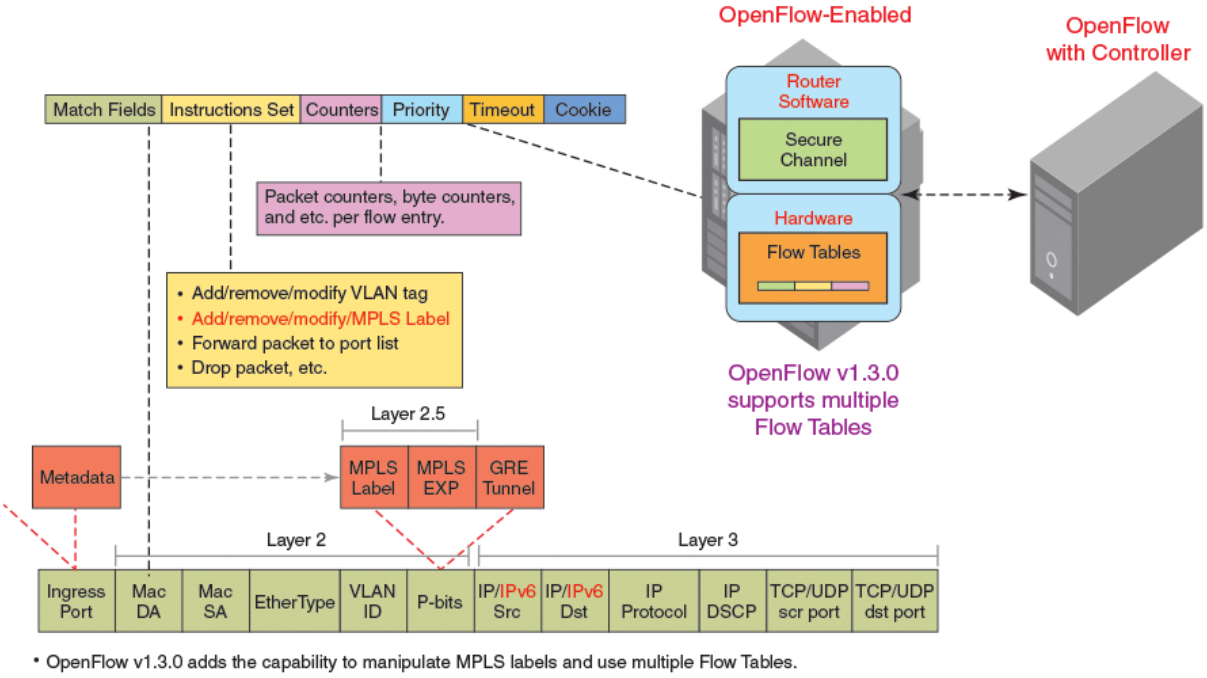
Izvor [26]

[7] SSL: engl. *Secure Sockets Layer,* standardni sigurnosna protokol za uspostavljanje šifrirane veze između klijenta i poslužitelja.

Važno je spomenuti kako *OpenFlow* specifikacija definira tri tipa tablica usmjeravanja. *Flow Table* podudara se s dolaznim paketima na određeni tok i određuje funkcije koje treba izvršiti na paketima. Može postojati više tablica toka koje djeluju na način cjevovoda (engl. *pipeline*), što će kasnije biti objašnjeno. *Flow Table* može usmjeriti tok tablice *Group Table*, što može izazvati različite radnje koje utječu na jedan ili više tokova podataka. *Meter Table* može pokrenuti niz radnji koje utječu na tok.

4.1. Tablica usmjeravanja Open Flow protokola

Osnovni dio logičke arhitekture *switch*-a je tablica usmjeravanja. Svaki paket koji ulazi u *switch* prolazi kroz jednu ili više tablica usmjeravanja, a svaka tablica usmjeravanja sadrži ulazne vrijednosti (Slika 8.).



Slika 8. Prikaz tablice usmjeravanja OpenFlow protokola

Izvor [27]

Ulazne vrijednosti sastoje se od šest komponenti:

- *Match Fields*: Koristi se za odabir paketa čije se vrijednosti u poljima podudaraju, odnosno odgovaraju jedna drugoj.
- *Priority*: Mogući prioriteti kod unosa tablice.
- *Counters*: Ažuriraju se za moguće podudaranje paketa. Specifikacija OpenFlow protokola definira različite vremenske oznake, npr. broj primljenih

bajtova i paketa po priključku, po tablici usmjeravanja ili po unosu tablice usmjeravanja, zatim broj ispuštenih paketa te trajanje usmjeravanja.

- *Instructions*: Aktivnosti koje treba poduzeti ako se paketi podudaraju.
- *Timeouts*: Maksimalno vrijeme mirovanja prije prekida toka.
- *Cookie*: Vrijednost podataka koju odabire kontroler. Može se koristiti za popunjavanje statistike toka ili izmjene i brisanje toka. Ne koristi se prilikom obrade paketa. [23]

4.1.1. Podudaranja tablica usmjeravanja

Tablica usmjeravanja može sadržavati ulazne vrijednosti propuštanja tablice, koje prikazuju sve oznake polja podudaranja i imaju najniži prioritet. Područje podudaranja polja ulaznih vrijednosti tablice usmjeravanja sastoji se od sljedećih obaveznih polja:

- *Ingress Port*: Identifikacija priključka na *switch*-u u koji je paket stigao. Može biti fizički priključak ili virtualni priključak koji definira *switch*.
- *Ethernet Source and Destination Addresses*: Svaka ulazna vrijednost može biti odgovarajuća adresa, vrijednost zapisana u bitovima, od kojih su samo neki zapisi provjereni ili se podudaraju sa svim vrijednostima.
- *IPv4 or IPv6 Protocol Number*: Vrijednost broja pprotokola, koja označava sljedeće zaglavlje u paketu.
- *IPv4 or IPv6 Source Address and Destination Address*: Svaka ulazna vrijednost može biti odgovarajuća adresa, vrijednost zapisana u bitovima, vrijednost podmreže ili zamjenska vrijednost.
- *TCP Source and Destination Ports*: Točno podudaranje ili zamjenska vrijednost.
- *UDP Source and Destination Ports*: Točno podudaranje ili zamjenska vrijednost.

Ranija polja podudaranja mora podržavati jedan od OpenFlow *switch*-eva, dok predstojeća polja mogu podržavati:

- *Physical Port*: Koristi se za određivanje osnovnog fizičkog porta na temelju poznavanja logičkog porta.
- *Metadata*: Dodatne informacije koje se mogu prenositi iz jedne tablice u drugu tijekom obrade podataka.

- *Ethernet Type*: Vrsta *Ethernet* priključka.
- *VLAN ID and VLAN User Priority*: Polja u zaglavlju IEEE 802.1Q virtualnog LAN-a (engl. *Local Area Network*).
- *IPv4 ili IPv6 DS i ECN*: Diferencirane usluge i polja obavjedti o zagušenju.
- *SCTP (engl. Stream Control Transmission Protocol) Source and Destination Ports*: Točno podudaranje ili zamjenska vrijednost.
- *ICMP (engl. Internet Control Message Protocol) Type and Code Fields*: Točno podudaranje ili zamjenska vrijednost.
- *ARP (engl. Address Resolution Protocol) Opcode*: Točno podudaranje u polju *Ethernet Type*.
- *Source and Target IPv4 Addresses in ARP Payload:* Svaka ulazna vrijednost može biti odgovarajuća adresa, vrijednost zapisana u bitovima, vrijednost podmreže ili zamjenska vrijednost.
- *IPv6 Flow Label:* Točno podudaranje ili zamjenska vrijednost.
- *ICMPv6 Type and Code fields*: Točno podudaranje ili zamjenska vrijednost.
- *IPv6 Neighbor Discovery Target Address*: otkriva susjedne poruke.
- *IPv6 Neighbor Discovery Source and Target Addresses*: opcije stvaranja veze za otkrivanje susjednih IPv6 poruka.
- *MPLS (engl. Multiprotocol Label Switching) Label Value, Traffic Class, and BoS (engl. Bottom of Stack):* polja u vrhu zaglavlja MPLS oznake.

Tako se OpenFlow specifikacije mogu koristiti s mrežnim prometom koji uključuje različite protokole i mrežne usluge. Važno je napomenuti da na podatkovnom sloju podržava samo *Ethernet*. Stoga, OpenFlow kao tako definiran ne može kontrolirati promet 2. sloja preko bežičnih mreža. Iz perspektive *switch*-a, prometni tok je niz paketa koji odgovara određenom ulazu u tablicu usmjeravanja. U tom slučaju, vrijednosti polja zaglavlja paketa tvore prometni tok, a ne ruta koju paketi slijede u mreži. Kombinacija ulaza toka na više *switch*-eva definira tok koji je vezan za određenu rutu. Komponenta *Instructions* tablice usmjeravanja, odnosno upute, sastoji se od niza uputa koje se izvršavaju ako paket odgovara ulazu. Prije opisivanja vrsta uputa, potrebno je definirati izraze *Action* i *Action Set*. Navedene komponente opisuju prosljeđivanje paketa, izmjenu paketa i postupke obrade *Group Table*-a. OpenFlow specifikacija uključuje sljedeće radnje:

- *Output*: prosljeđuje paket na određeni ulaz.

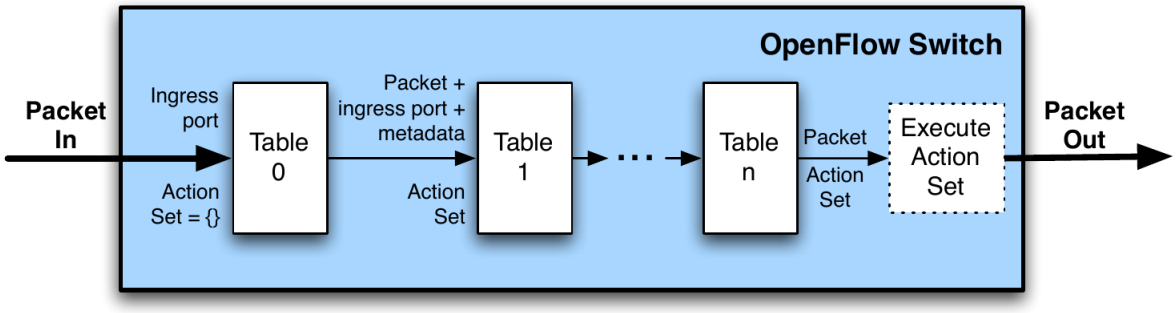
- *Set-Queue*: Postavlja ID čekanja za paket. Pri dolasku paketa na switch, ID čekanja određuje koji će se red čekanja koristi za preuzimanje i prosljeđivanje paketa. Proslijeđivanje je definirano konfiguracijom reda i koristi se za pružanje osnovne QoS podrške.
- *Group*: Obrađeni paket u određenoj skupini.
- *Push-Tag / Pop-Tag*: Polje oznake za VLAN ili MPLS paket.
- *Set-Field*: Razne *Set-Field* radnje identificirane su po vrsti polja; one mijenjaju vrijednosti odgovarajućih polja u zaglavlju paketa.
- *Change-TTL*: Različite *Change-TTL*[8] akcije mijenjaju vrijednosti određenih, vremenski ovisnih polja u zaglavlju paketa.

Action Set je popis radnji koje se dodjeljuju paketu za vrijeme obrade prema tablicama usmjeravanja i izvršavaju se kada paket izađe iz cjevovoda. Razlikujemo četiri vrste uputa:

- *Direct packet through pipeline*: Uputa *Goto-Table* usmjerava paket na sljedeću tablicu u cjevovodu. Uputa *Meter* usmjerava paket na određeni mjerač.
- *Perform action on packet*: Akcije se mogu izvršiti na paketu kada se podudara s ulaznim jedinicama tablice.
- *Update action set*: Unos određenih radnji u *Action Set* za ovaj paket ili brisanje svih radnji iz *Action Set*-a.
- *Update metadata*: Vrijednosti metapodataka mogu se povezati s paketom. Koriste se za prijenos podataka iz jedne tablice u drugu. [23]

4.1.2. Cjevovod tablica usmjeravanja

Switch uključuje jednu ili više tablica usmjeravanja. U slučaju da postoji više od jedne tablice usmjeravanja, organiziraju se u cjevovod (Slika 9.).



[8] TTL: engl. *Time To Live,* mehanizam koji ograničava vrijeme zadržavanja paketa u mreži.

Slika 9. Flow – table pipeline

Izvor [28]

Kada je paket prikazan u tablici podudaranja, ulaz se sastoji od paketa, identiteta ulaza, pridružene vrijednosti metapodataka i pripadajućeg *Action Set*-a. Za tablicu 0, vrijednost metapodataka je prazna i radnja je nula. Obrada se nastavlja sljedećim redosljedom:

1. Potrebno je pronaći ulaznu jedinicu najvećeg prioriteta podudaranja. Ako se ne podudara ni s jednim ulazom i ne postoji ulaz za propuštanje tablice, paket se ispušta. Ako se podudaraju samo na ulazu propuštanja tablice, tada taj ulaz određuje jednu od tri radnje:
   a) Slanje paketa na kontroler. Ova radnja kontroleru omogućuje definiciju novog toka za slične pakete ili ispuštanje paketa.
   b) Prosljeđivanje paketa prema sljedećoj tablici u cjevovodu.
   c) Ispuštanje paketa.
2. Ako se podudara s jednim ili više ulaza, osim ulaza u tablicu, to se podudaranje definira i kao ulaz s najvećim prioritetom podudaranja. Tada se izvršavaju sljedeće radnje:
   a) Ažuriranje svih brojila povezanih s ovim ulazom.
   b) Izvršavanje svih uputa povezanih s ovim ulazom. Te upute mogu uključivati ažuriranje *Action Set*-a, ažuriranje vrijednosti metapodataka i izvođenje radnji.
   c) Paket se zatim prosljeđuje prema sljedećoj tablici u cjevovodu. Može biti usmjeren na *Group Table*, *Meter Table* ili prema izlaznom portu.

Posljednja tablica usmjeravanja u cjevovodu nema opciju prosljeđivanje na drugu tablicu usmjeravanja. Kada se paket konačno usmjeri na izlazni port, izvršava se dodijeljeni *Action Set*, a paket čeka na izlaz. [23]

4.2. Poruke Open Flow protokola

OpenFlow protokol može se opisati i kao protokol za razmjenu poruka između OpenFlow kontrolera i OpenFlow *switch*-eva. Protokol se implementira u vrhu SSL-a

ili TLS[9]-a (engl. *Transport Layer Security*), pružajući siguran OpenFlow kanal. OpenFlow protokol omogućuje kontroleru dodavanje, ažuriranje i brisanje podataka u tablicama usmjeravanja. Podržane su tri vrste poruka, kao što je prikazano u tablici 2.:

- *Controller-to-Switch:* Ove poruke pokreće kontroler i zahtjeva odgovor od *switch*-a. Omogućuju kontroleru upravljanje logičkom izvedbom *switch*-a, uključujući konfiguraciju i detalje unosa tablice usmjeravanja. U ovu klasu poruka uključena je i poruka *Packet-out*, koja se koristi u slučaju kada *switch* šalje paket kontroleru, a kontroler odluči usmjeriti paket izravno na izlaznu priključnicu *switch*-a.
- *Asynchronous*: Ove poruke šalju se bez upravljanja kontrolera. Ova klasa poruka sadrži različite statusne poruke usmjerene kontroleru. Također je uključena i poruka *Packet-in*, koju *switch* može koristiti za slanje paketa kontroleru kada nema podudaranja ulaznih jednica tablice usmjeravanja.
- *Symmetric*: Ove poruke šalju se bez upravljanja kontrolera ili *switch*-a te su jednostavnog i korisnog karaktera. *Hello* poruke se obično šalju izmjenično između kontrolera i *switch*-a kada je veza uspostavljena. *Echo* poruke zahtjeva i odgovora mogu se koristiti uz pomoć *switch*-a ili kontrolera za mjerenje latencije i pojasne širine ili samo kako bi se provjerila ispravnosot rada uređaja. Eksperimentalne poruke koriste se za definiranje značajki koje će biti ugrađene u buduće inačice OpenFlow protokola i njegov temeljni model. Svaki *switch* održava relacijski model podataka koji sadrži atribute za svaku OpenFlow apstrakciju. Ti atributi mogu opisivati sposobnost apstrakcije, stanje konfiguracije ili neki skup aktualnih statistika. [23]

Tablica 2. Poruke OpenFlow protokola

Poruka	Opis
Controller-to-Switch	
Features	Provjera mogućnosti *switch*-a. *Switch* odgovara povratom koji određuje njegove

[9] TLS: engl. *Transport Layer Security*, mrežni protokol zadužen za informacijsku sigurnost u odnosu između klijenta i poslužitelja.

	mogućnosti.
Configuration	Postavljanje i provjera konfiguracijskih parametara. *Switch* odgovara postavkama parametara.
Modify-State	Dodavanje, brisanje ili izmjena unosa toka / grupa i postavljanje svojstava portova *switch*-a.
Read-State	Prikupljanje podatke iz switch-a, kao što su trenutna konfiguracija, statistika i mogućnosti.
Packet-out *Barrier*	Izravno prosljeđivanje paketa prema određenom portu *switch*-a. Kontroler koristi upit zahtjev / odgovor na barijeru kako bi se definirala međuovisnost poruka ili kako bi primio obavjest o izvršenim radnjama.
Role-Request	Postavljanje ili provjera uloge OpenFlow kanala. Korisno kada se jedan *switch* povezuje s više kontrolera.
Asynchronous-Configuration	Postavljanje ili upis filtera za asinkronizirane poruke. Korisno kada se jedan *switch* povezuje s više kontrolera.
Asynchronous	
Packet-in	Prosljeđivanje paketa kontroleru.
Flow-Removed	Obavijest kontroleru o uklanjanju unosa prometnog toka iz tablice usmjeravanja.
Port-Status	Obavijest kontroleru o promjeni na portu.
Error	Obavijest kontroleru o greškama ili problemima.
Symmetric	
Hello	Obavijest o uspostavi veze između

	kontrolera i *switch*-a.
Echo	Zahtjevanje povratne informacije, mogu se koristiti i sa kontrolera i sa *switch*-a.
Experimenter	Za upotrebu dodatnih funkcija.

Izvor [23]

U nastavku slijedi opis *OpenFlow* zaglavlja, koje je sastavni dio svake *OpenFlow* poruke.

4.3. Zaglavlje OpenFlow protokola

Svaka *OpenFlow* poruka počinje *OpenFlow* zaglavljem, koje se sastoji od sljedećih linija koda:

```
/* Header on all OpenFlow packets. */
struct ofp_header {
        uint8_t version; /* OFP_VERSION. */
        uint8_t type; /* One of the OFPT_ constants. */
        uint16_t length; /* Length including this ofp_header. */
        uint32_t xid; /* Transaction id associated with this packet.
                         Replies use the same id as was in the request
                         to facilitate pairing. */
    };
    OFP_ASSERT(sizeof(struct ofp_header) == 8);
```

Ova verzija zaglavlja predstavlja verziju koja se koristi kod OpenFlow protokola. Tijekom ranije faze razvoja OpenFlow protokola, najvažniji je bit koji označava eksperimentalnu verziju, a niži bitovi označavaju potrebne ispravke. Verzija protokola opisana ovom specifikacijom nosi naziv 1.4.0, a njegova inačica je 0x05. Polje duljine označava ukupnu duljinu poruke, stoga nije potreban zaseban okvir za razlikovanje jednog okvira od drugog. Vrste mogu imati sljedeće vrijednosti:

```
enum ofp_type {
        /* Immutable messages. */
        OFPT_HELLO = 0, /* Symmetric message */
        OFPT_ERROR = 1, /* Symmetric message */
        OFPT_ECHO_REQUEST = 2, /* Symmetric message */
```

```
OFPT_ECHO_REPLY = 3, /* Symmetric message */
OFPT_EXPERIMENTER = 4, /* Symmetric message */
/* Switch configuration messages. */
OFPT_FEATURES_REQUEST = 5, /* Controller/switch message */
OFPT_FEATURES_REPLY = 6, /* Controller/switch message */
OFPT_GET_CONFIG_REQUEST = 7, /* Controller/switch message */
OFPT_GET_CONFIG_REPLY = 8, /* Controller/switch message */
OFPT_SET_CONFIG = 9, /* Controller/switch message */
/* Asynchronous messages. */
OFPT_PACKET_IN = 10, /* Async message */
OFPT_FLOW_REMOVED = 11, /* Async message */
OFPT_PORT_STATUS = 12, /* Async message */
/* Controller command messages. */
OFPT_PACKET_OUT = 13, /* Controller/switch message */
OFPT_FLOW_MOD = 14, /* Controller/switch message */
OFPT_GROUP_MOD = 15, /* Controller/switch message */
OFPT_PORT_MOD = 16, /* Controller/switch message */
OFPT_TABLE_MOD = 17, /* Controller/switch message */
/* Multipart messages. */
OFPT_MULTIPART_REQUEST = 18, /* Controller/switch message */
OFPT_MULTIPART_REPLY = 19, /* Controller/switch message */
/* Barrier messages. */
OFPT_BARRIER_REQUEST = 20, /* Controller/switch message */
OFPT_BARRIER_REPLY = 21, /* Controller/switch message */
/* Controller role change request messages. */
OFPT_ROLE_REQUEST = 24, /* Controller/switch message */
OFPT_ROLE_REPLY = 25, /* Controller/switch message */
/* Asynchronous message configuration. */
OFPT_GET_ASYNC_REQUEST = 26, /* Controller/switch message */
OFPT_GET_ASYNC_REPLY = 27, /* Controller/switch message */
OFPT_SET_ASYNC = 28, /* Controller/switch message */
```

```
        /* Meters and rate limiters configuration messages. */
        OFPT_METER_MOD = 29, /* Controller/switch message */
        /* Controller role change event messages. */
        OFPT_ROLE_STATUS = 30, /* Async message */
        /* Asynchronous messages. */
        OFPT_TABLE_STATUS = 31, /* Async message */
        /* Request forwarding by the switch. */
        OFPT_REQUESTFORWARD = 32, /* Async message */
        /* Bundle operations (multiple messages as a single operation). */
        OFPT_BUNDLE_CONTROL = 33,
        OFPT_BUNDLE_ADD_MESSAGE = 34,
};
```
[27]

Većina OpenFlow poruka sadrži polja ispune. Ona su uključena u razne vrste poruka i zajedničke strukture. Većina tih polja ispune jednoznačno je označena izrazom *pad* u početku zapisa. Cilj polja ispune je poravnanje entiteta većih veličina unutar granica procesorske moći. Sve uobičajene strukture poruka poravnavaju se 64 bitnim granicama, dok su rjeđe vrste poruka po potrebi poravnate 32-bitnim granicama. [29]

5. Planiranje SDN mreža

Telekomunikacijskim sustavom se ostvaruje prenošenje informacija do korisnika koji primjećuju veću vrijednost u poslovanju zbog pristupačnosti informacija. Dobro planiranje zahtijeva primjenu znanstvenih metoda, a time i izvođenje planiranja treba izvoditi na visokoj stručnoj razini, uz najužu koordinaciju tehnike i tehnologije te gospodarstva i financijskih institucija. Kako bi se smanjio rizik odstupanja od ciljeva projekta, važno je predviđanje svih mogućih čimbenika i što duže praćenje ciljeva projekta, bez obzira na neizvjesnost projekta. [30]

Struktura i sadržaj planiranja u telekomunikacijskoj mreži mora se temeljiti na:

- analizi postojećeg stanja u telekomunikacijskoj mreži koja mora obuhvaćati instalirane priključne točke, stupanj iskorištenosti postojeće infrastrukture, te funkcionalnost iste.
- planovima proširenja postojećih telekomunikacijskih mreža, razvojnim planovima usluga u telekomunikacijskoj mreži i uvođenjem novih aplikacija kao podrške u implementiranju novih usluga.
- ekonomskoj opravdanosti ciljanog zahvata tj. analize troškova, rashoda, prihoda i povrata uloženih sredstava.

Na planiranje telekomunikacija utječu sljedeći unutarnji i vanjski čimbenici:

- Unutarnji čimbenici: organizacija, struktura kadrova, tehnička i tehnološka opremljenost i sl.
- Vanjski čimbenici: uvjeti privređivanja, gospodarstvo, cijene i sl.

U postupku planiranja mreže razlikuje se nekoliko vrsta planova:

- Planovi s obzirom na vremensko razdoblje:
    - kratkoročni planovi za razdoblje do jedne godine,
    - srednjoročni planovi za razdoblje od pet godina,
    - dugoročni planovi za razdoblje od deset pa i više godina.
- Planovi s obzirom na predmet planiranja:
    - ukupni planovi za plan razvoja ili plan poslovanja,
    - pojedinačni koji mogu biti:
        - naturalni (plan kapaciteta, radne snage ili usluga),

        - financijski (plan cijena, plan prihoda, plan troškova).
    - Planovi s obzirom na područje zahvata:
        - državni,
        - regionalni(županijski),
        - lokalni(općinski),
        - poslovni.

Prije samih faza planiranja potrebno je odrediti organizaciju planiranja koja može biti:

- centralizirana (kod manjih organizacija),
- decentralizirana (kod srednjih organizacija),
- kombinirana (kod velikih organizacija).

Faze planiranja mogu se podijeliti u tri dijela:

- priprema; koja podrazumijeva prikupljanje podataka o dosadašnjem stanju veličine za koju se planira te utjecaju čimbenika na taj razvoj.
- Izradba; koja se sastoji od analize dosadašnjeg razvoja veličine za koju se planira, utjecaja čimbenika na razvoj , postavljanja ciljeva koji se žele ostvariti i plana najprikladnijih putova za njihovo ostvarenje.
- donošenje planova; odnosno određivanje poželjnih rokova pa prema tome možemo imati kratkoročne planove (donose se do mjesec dana prije planiranog razdoblja), srednjoročni planovi (donose se šest mjeseci prije početka planskog razdoblja) i dugoročni planovi koji se donose i do godinu dana prije početka planiranog razdoblja.

Tipovi planiranja u telekomunikacijskim mrežama:

- strategijsko,
- postojeće,
- operativno.

Postoje tri tzv. „pogleda“ telekomunikacijskog sustava: *cost center, resource* i *strategic weapon*. Telekomunikacije se tako može promatrati kroz međuodnos organizacije i jednog od tri navedena pogleda:

- troškovi se moraju minimalizirati,

- programi poboljšavaju produktivnost zaposlenika (radne snage),
- strateško oružje, koje se u stjecaju prilika, sa informacijskim sustavom tvrtke, koristi za poboljšanje njezinog konkurentnog položaja te otvara nove mogućnosti.

Pojavljuju se nova područja kojima je potrebno posvetiti veliku pozornost:

- marketing - potrebno je kreirati koncepciju i strategiju marketinga koja će operatoru osigurati konkurentsku prednost, tj. kojom će preduhitriti konkurencije u ponudi usluga i paketa usluga korisnicima,
- korisnički zahtjevi - u kreiranju ponude svojih usluga/proizvoda operator nužno,
- mora osluškivati zahtjeve svojih poslovnih i privatnih korisnika,
- tehnologija,
- standardi,
- informatička podrška - nužnom se nameće potreba uporabe najnovijih IT rješenja kako bi se omogućilo što lakše planiranje i proširenje kapaciteta,
- troškovi - planeri se suočavaju s velikim brojem planskih metoda, pri čemu su uključene različite opcije i parametri. Strategija svakog operatora je promjenjiva što u velikoj mjeri može utjecati na investicije (arhitektura, rezerve, tehnologija...). [30]

Nakon točno definiranih svih aspekata planiranja mreže, jasno je kako je implementacija SDN-a, bez edukacije sudionika u procesu, određeni rizik. Također i samo ignoriranje SDN-a predstavlja rizik za telekomunikacijske organizacije, kao i za stručnjake u području telekomunikacija. Rizik za telekomunikacijske organizacije je u tome što neće biti u stanju riješiti probleme za koje je SDN u prvom redu dizajniran, čega je posljedica nedostatak konkurentnosti na tržištu. Rizik za stručnjake u području telekomunikacija je nedostatak edukacije vezane za ovaj način umrežavanja, a samim tim neće biti konkurentni na tržištu rada. Na temelju analiza i istraživanja može se zaključiti da će SDN u narednih nekoliko godina imati značajan utjecaj i na mreže poduzeća i na uloge samih mrežnih stručnjaka. Zbog toga se mora razviti plan za implementaciju SDN-a na telekomunikacijskom tržištu. S obzirom

na brze promjene u tehnološkom razvoju, pa tako i razvoju SDN-a, bilo koji plan implementacije razvijat će se dalje tokom vremena.

Proučavajući dostupne izvore i istraživanja stručnjaka, može se naići na brojna rješenja koji opisuju postupke planiranja implementacije SDN-a, u nastavku su opisani koraci primjera takvog rješenja:

1. Definiranje SDN-a,
2. Prepoznavanje primarnih mogućnosti,
3. Prepoznavanje ključnih udaljenosti,
4. Odlučivanje,
5. Procjena SDN rješenja,
6. Ispitivanje i certificiranje rješenja,
7. Integracija s postojećom okolinom,
8. Educiranje osoblja,
9. Ocjenjivanje profesionalnih usluga,
10. Uklanjanje otpora organizacije,
11. Izvršavanje POC-a (engl. *Proof of Concept*),
12. Dobivanje MBI-a (engl. *Management Buy-In*). [31]

Naravno, proces implementacije uvelike ovisi o veličini i složenosti mreže te iskustvu i stupnju edukacije sudionika. Potrebne su nove vještine i dodatna edukacija, s obzirom da prijelaz s tradicionalne na softverski definiranu unosi brojne promjene. Uz dobro planiranje, većina organizacija će brzo uvidjeti i iskoristiti sve očigledne performanse SDN rješenja. Među telekomunikacijskim stručnjacima postalo je uvriježeno misliti kako je SDN u trenutku pojave obećavao puno, ali kasnije zapravo nikad nije zaživio. Ipak, nema sumnje da se SDN zapravo drži u industriji. Telekomunikacijske organizacije vodile su naplatu, ali sada tvrtke također pronalaze načine implementacije SDN-a u rješavanju različitih izazova s kojima se suočavaju. Tvrtke međusobno dijele svoja iskustva, što pozitivno utječe na jednostavnost edukacije. Uspješna implementacija SDN-a u poduzeću zahtijeva fokusiranje na slučajeve uporabe koje su testirali drugi i za izradu jasnih planova. Ukoliko nije jasno definiran problem koji se može riješiti jednim od slučajeva uporabe SDN platforme, moguće je naći vrlo brz put do uspjeha. Uvođenje SDN-a u poslovanje „polovično“,

točnije, bez korištenja konkretnih slučajeva uporabe i uključenja svih sudionika neće polučiti rezultate. Činjenica je da SDN povećava složenost, a uz složenost dolazi apstrakcija na višoj razini. Operativne složenosti postaju nešto za što je potrebno postići kompromis. Sposobnost podrške SDN platformi može biti izazov za tvrtke, ali budući da dolazi do usvajanja i inovacija SDN-a, taj je problem sve manje prisutan. Na izazov se nailazi i u trenutku kada je ponuđeno previše rješenja. Mnoge organizacije, uzimajući u obzir izazov posjedovanja i vođenja SDN platforme, imaju problema s korištenjem konkretnih slučajeva uporabe. Zato je razumijevanje slučajeva uporabe važno za rješavanje složenijih problema SDN platforme.

Jednom kada su definirani svi slučajevi uporabe, kako bi se uspješno implementirao SDN, stručnjaci savjetuju sljedeće:

- Pronalazak partnera – Preporuča se iskoristiti usluge i ponude proizvođača opreme te ih upotrijebiti u svrhu lakšeg pokretanja projekta.
- Konferencije – S obzirom da je prisutan velik interes za određena područja, velika je i prilika za učenje. Međutim, ako ne postoji mogućnost osobnog prisustvovanja skupovima stručnjaka, mnogi dijele sadržaj svojih predavanja na mreži.
- Edukacija osoblja - Nema nedostatka SDN resursa koji nisu otkriveni i za koje ne postoji adekvatna edukacija. Razvijanje SDN plana s obrazovanim timom uvelike će povećati uspjeh implementacije.
- Laboratoriji - Vrijednost SDN-a je i u tome što ga je moguće implementirati pomoću virtualnih laboratorija, kako na web-lokaciji tako i u okruženjima koja se temelje na računalstvu u oblaku. Većina SDN proizvoda dostupna je u okruženju ispitnog laboratorija zahvaljujući činjenici da je softver prijenosan.
- Praćenje konkurencije – Praćenjem konkurencije na telekomunikacijskom tržištu te poznavajući opremu i metode koje koriste u sličnim slučajevima uporabe, omogućeno je brže djelovanje na problematiku konkretnog slučaja uporabe.

Gledano iz aspekta koncepata razvoja SDN-a, pred upravljanje mrežom postavljaju se određeni zahtjevi i izazovi. Odnoseći se na sve entitete u mreži, neki od najvažnijih zahtjeva i izazova su:

- **Pokretanje sustava i konfiguracija**; Kod tradicionalnih mreža, upravljanje mrežom i prosljeđivanje nalazi se unutar svakog mrežnog uređaja zasebno i u

skladu sa skupom standardiziranih protokola. Razdvajanjem upravljačkog dijela mreže i podataka kod SDN-a, pojavljuje se potreba za međusobnom komunikacijom ta dva dijela mreže. Konfiguriranje njihove komunikacije može biti vrlo složeno, s obzirom da i jedan i drugi dio mogu upotrebljavati softverski definirane protokole. Dakle, bilo kakve promjene software-a u upravljačkom dijelu mogu izravno utjecati na dio prosljeđivanja i obratno. U savršenim uvjetima, softverski definirani protokoli trebali bi imati vlastito upravljačko sučelje kako bi se omogućilo pravilno pokretanje i konfiguriranje, ali i vraćanje u početno stanje i lakše praćenje i održavanje sustava.

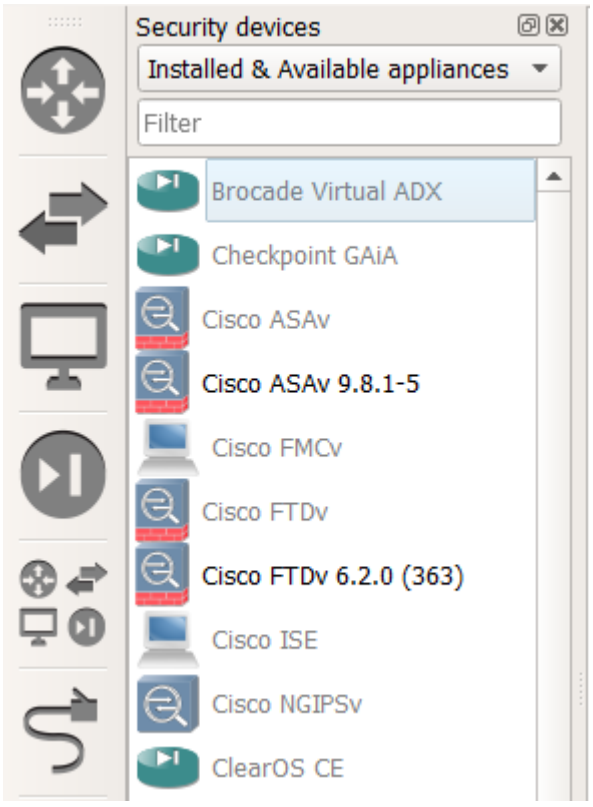
- **Dostupnost i otpornost**; SDN je vrlo osjetljiv na pogreške kako u softverskim tako i u hardverskim entitetima. Pogotovo ako se radi o mogućem prekidu komunikacije između upravljačkog dijela mreže i dijela prosljeđivanja, odnosno između kontrolera i *switch*-eva. U nekim konceptima SDN-a razmatra se i postavljanje upravljačke jedinice unutar jednog čvora, ali u tom slučaju postavlja se pitanje dostupnosti i otpora zbog jedne točke neuspjeha (engl. *Single Point of Failure*, SPOF). Usprkos tome što su *switch*-evi u stanju odlično odgovoriti na ovakvu problematiku, vrlo je važno u skladu s pravilima upravljati i konstantno održavati vezu između upravljačkog dijela mreže i dijela prosljeđivanja.
- **Mrežno programiranje**; Sa stajališta mrežnih administratora SDN-a, svako puštanje novog ili ažuriranje *software*-a mora se primjeniti na sve dijelove mreže jednako. Kako bi se osigurala odgovarajuća podrška za upravljanje mrežnim programiranjem, potrebni su mrežni alati koji omogućuju mrežnim administratorima kontrolu verzije *software*-a, koordiniranu implementaciju, povrat na početno stanje i provjeru mrežnog software-a. Dakle, software za određivanje ponašanja mreže može se također razvijati u različitim razinama apstrakcije. Svakako je potrebno da su alati i metode dizajnirani na način da razdvajaju pravila i naredbe visoke ili srednje razine od konfiguracije mrežnih uređaja koji se nalaze na najnižoj razini.
- **Performanse i skalabilnost**; Kod softverski definiranog umrežavanja mrežni *hardware* mora biti generički kako bi *software* mogao pripadati visokim razinama apstrakcije. Dakle, odvajanje upravljačkog dijela mreže od podatkovnog podrazumijeva dodatne komunikacijske zahtjeve između dijelova mreže, koji vrlo često mogu uzrokovati kašnjenja. Velika odgovornost za

osiguranje učinkovitosti dodjeljuje se razvojnim inženjerima od kojih se očekuje optimizacija *software*-a. Ostatak odgovornosti dodjeljen je mrežnim administratorima od kojih se očekuje uskladiti parametre za optimizaciju softverski definiranih protokola sa najučinkovitijim kontrolnim i upravljačkim modelima. [32].

Zaključno, performanse SDN-a nisu vidljive „preko noći“. Kao što je implementacija tradicionalnih mreža, do svakog kako privatnog tako i poslovnog korisnika, trajala dugi niz godina, tako je potrebno vrijeme prilagodbe i usvajanja SDN koncepata. Pozitivno je što se znanja šire među konkurencijom i tržište je sve otvorenije inovacijama.

6. Prikaz konfiguracije i simulacije SDN mreža u programskom alatu GNS3

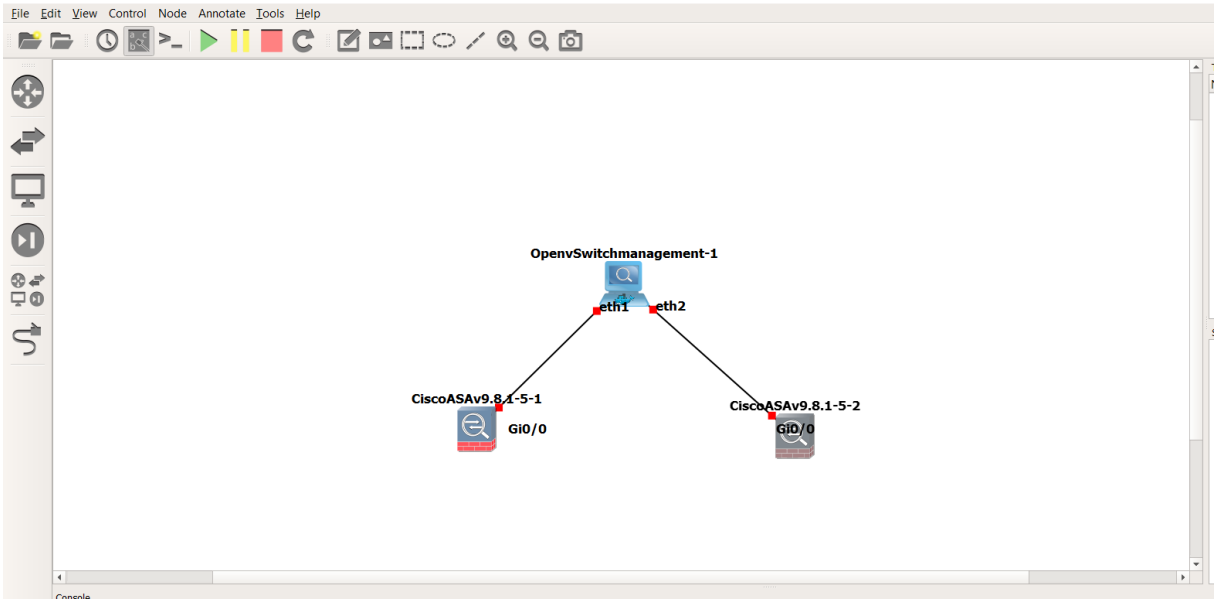
GNS3 (engl. *Graphical Network Simulator-3*) je grafički mrežni simulator koji omogućuje dizajniranje složenih mrežnih topologija. Moguće je pokrenuti simulacije ili konfiguracije uređaja u rasponu od najjednostavnijih terminala do složenih *routera*. Ovo poglavlje sastoji se od opisa simulatora te simulacije Mininet[10] SDN platforme u GNS3, te mjerenja osnovnih mrežnih parametara kao što su dostupnost i propusnost, odnosno usporedbe tradicionalne mreže i mreže koja je softverski oblikovana. U nastavku slijedi prvo opis GNS3 simulatora te njegovih komponenti.



Slika 10. Izbornik emuliranih uređaja

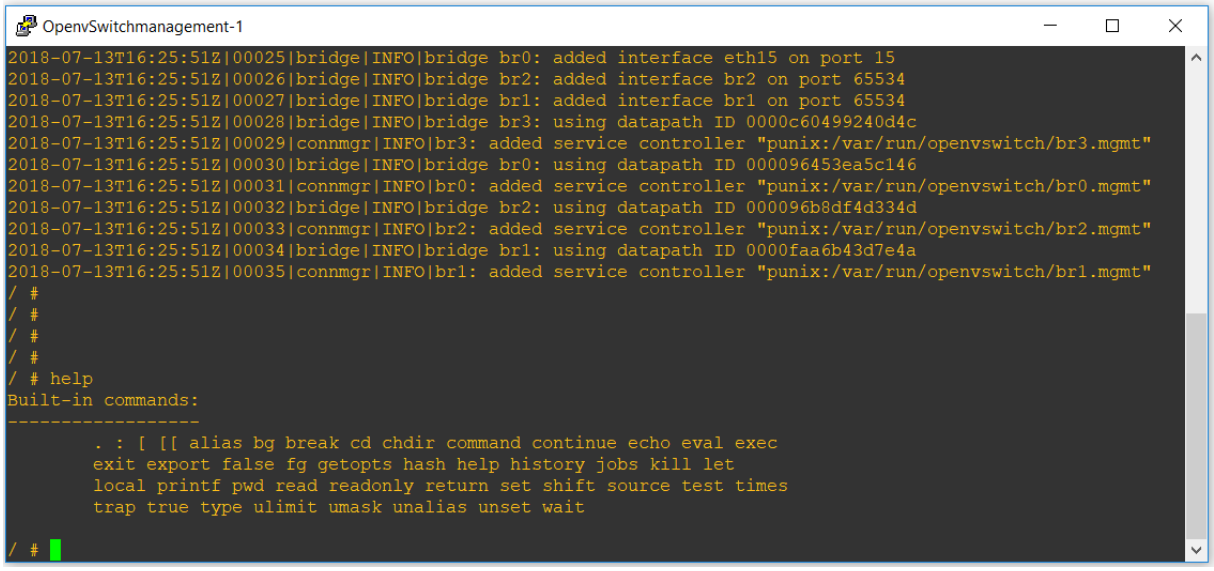
[10] Mininet: platforma koja pruža virtualni testni prostor i razvojno okruženje za softverski definirane mreže.

Slikom 10 prikazan je po kategorijama izbornik uređaja koje je GNS3 simulator u stanju pokrenuti. Radi se o raznim preklopnicima, usmjernicima, uređajima za zaštitu sustava te terminalnim uređajima.



Slika 11. Radna površina GNS3 emulatora

Dodavanje spomenutih uređaja obavlja se tzv. „*drag and drop*“ metodom na radnu površinu (Slika 11.) emulatora.

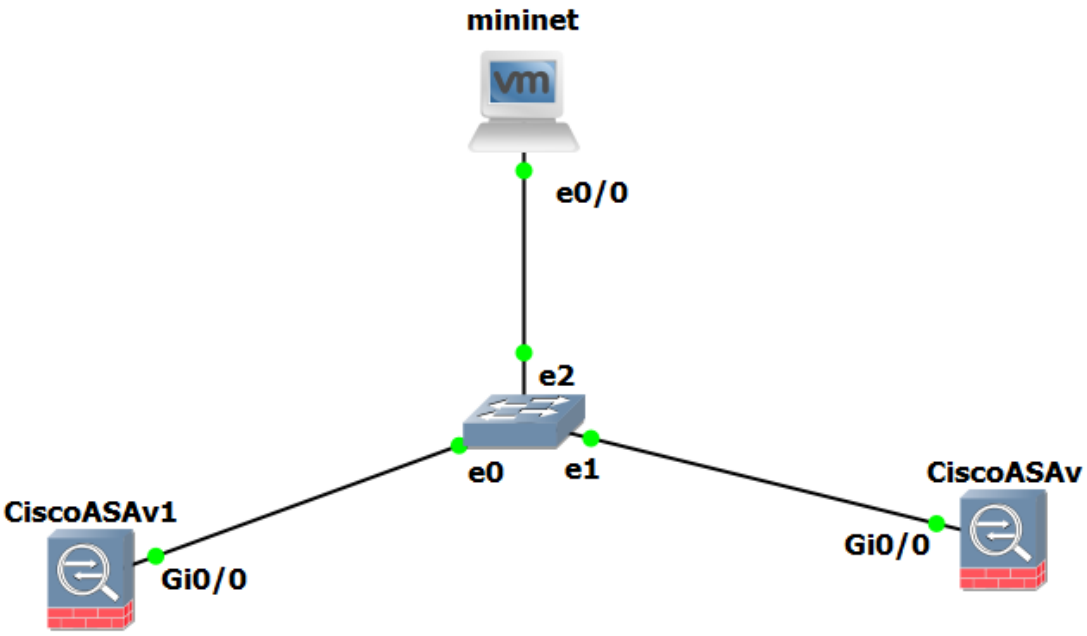


Slika 12. Terminal (CLI)

Uređaji se mogu pokrenuti pojedinačno ili skupno, dvostrukim klikom na odabrani uređaj otvara se terminal (Slika 12.), odnosno CLI (engl. *Command Line Interface*).

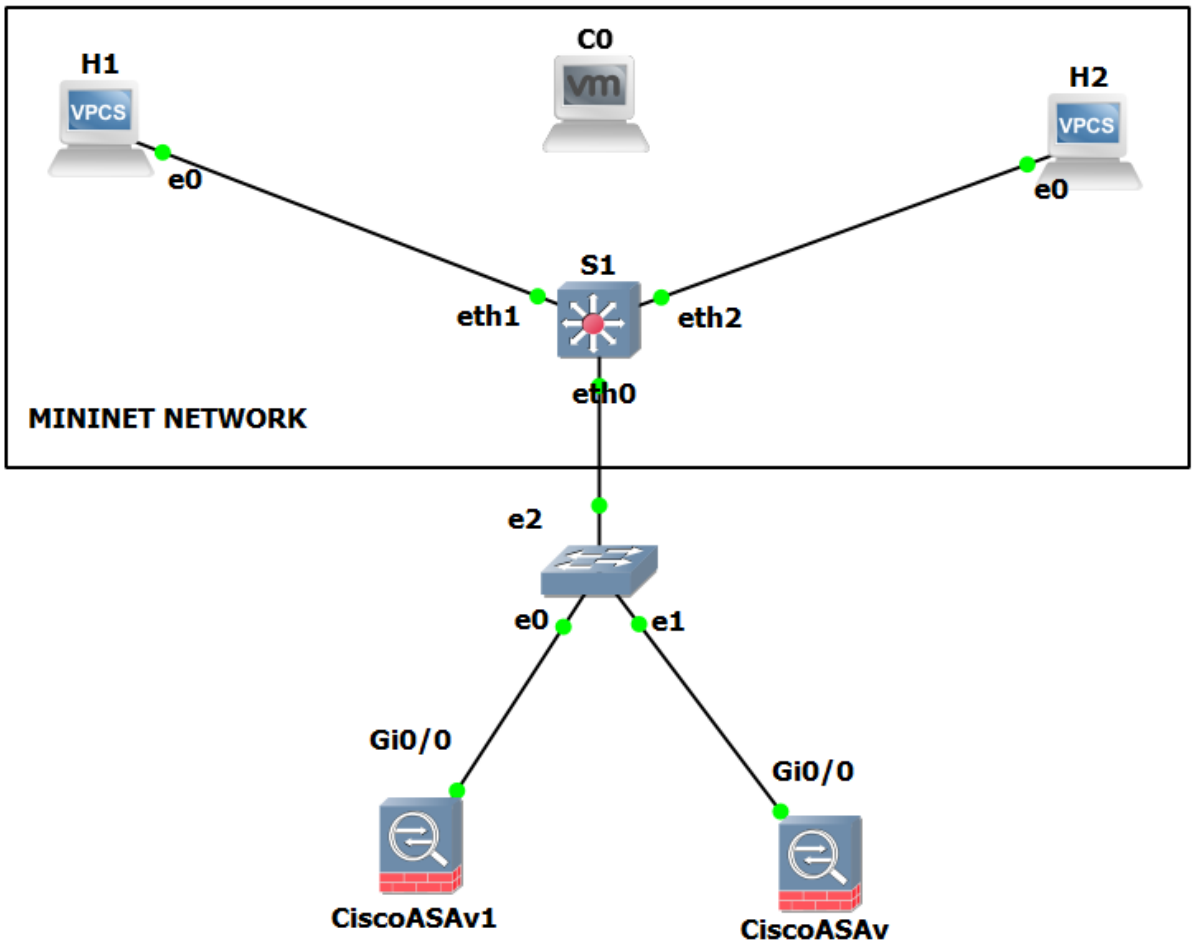
6.1 Simulacija Mininet SDN platforme

Mininet je platforma bazirana na *Linux*[11]-u koja omogućuje definiranje i upravljanje virtualnim mrežnim objektima, kao što su preklopnici, usmjernici i dr., a isto tako za ulogu ima nadzor komunikacije između svih virtualnih komponenti, a to je moguće jer u sebi sadrži *OpenFlow* protokol pomoću kojeg šalje naredbe prema komponentama unutar sustava. Za potrebe simulacije bit će korišteni tradicionalni vatrozidi, u sklopu mininet platforme. Kreirat će se jedan virtualni preklopnik (S1), dva terminalna uređaja (H1 i H2) (engl. *host*) te jedan SDN kontroler (C0). U nastavku slijedi fizički (Slika 13.) i logički (Slika 14.) prikaz sustava.



Slika 13. Fizički prikaz simuliranog sustava

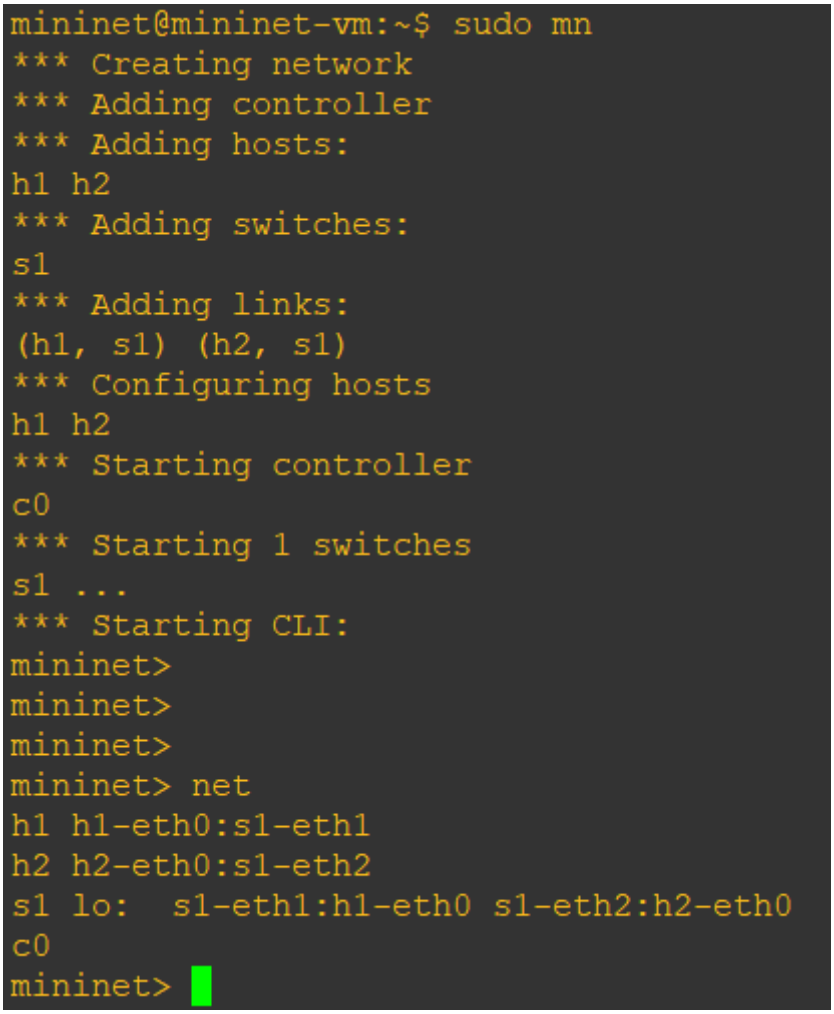
[11] Linux: najpoznatiji i najčešće korišten operacijski sustav otvorenog koda. Kao operativni sustav, Linux je softver koji predstavlja temelj svih ostalih softvera na računalu, prima zahtjeve tih programa i prenosi te zahtjeve na hardver računala.



Slika 14. Logički prikaz simuliranog sustava

Pregledom Slike 13. i Slike 14. može se zaključiti kako se Mininet mrežom može postići kontrola sustava visoke razine, odnosno kako bi se ista konfiguracija primijenila na tradicionalnim vatrozidima, potreban je pristup svakom uređaju posebno te bi se tim uređajima upravljali zasebno, odnosno ti uređaji bi imali odvojen *managament plane*. Koristeći Mininet, mrežni uređaji i ostale komponente dijele jedan *management plane*, a kontrola se postiže na način da se uređajima upravlja s jednog terminala.

Slijedi simulacija te prikaz naredbi za kreiranje Mininet mreže (Slika 15).



Slika 15. Kreiranje mininet mreže

Naredba za kreiranje mininet mreže:

mininet@mininet-vm:~$ sudo mn

*** Creating network

*** Adding controller

*** Adding hosts:

h1 h2

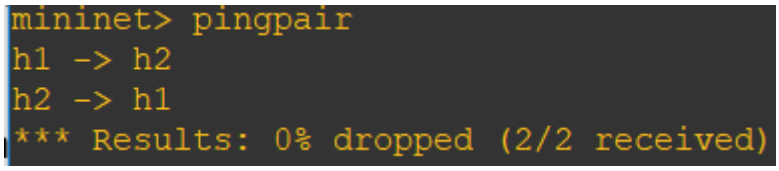
```
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
```

Naredba za provjeru topologije mreže:

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
```

6.2 Mjerenje osnovnih mrežnih parametara SDN mreže

Za osnovne mrežne parametre uzeta su ključna mjerila performansi: propusnost, gubitak paketa, kašnjenje i *jitter*, o kojima će se govoriti u sljedećem poglavlju. Način njihovog mjerenja će se odrediti putem SDN kontrolera (kratica C0) u mininet mreži. Gubitak paketa će se izmjeriti preko naredbe *ping* (Slika 16.) između dva terminalna uređaja u mininet mreži, a to su H1 i H2 te između svih komponenti sustava, bitno je skrenuti pozornost na način izvedbe naredbe te same provjere dostupnosti uređaja.



Slika 16. Provjera gubitka paketa

Naredba za provjeru dostupnosti između H1 i H2:

mininet> pingpair
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet>

Naredba za provjeru dostupnosti između svih komponenti:

mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
h1 -> h3
h3 -> h1
h2 -> h3
h3 -> h2
*** Results: 0% dropped (2/2 received)

U svrhu usporedbe rezultata dodan je još jedan terminalni uređaj (H3) u mininet mrežu. Kada bi se htjeli dobiti isti rezultati u tradicionalnoj mreži te ako se za primjer uzmu vatrozidi iz opisanog sustava, potrebno je napraviti slijedeće radnje:

1. Spojiti se na svaki vatrozid posebno,

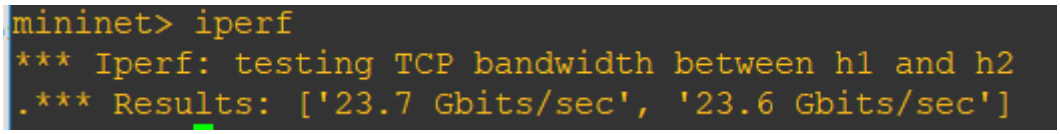
2. Upisati naredbu „ping“ prema i od svakog vatrozida.

Ukoliko se radi o upravljanju dvaju vatrozida to ne predstavlja problem, ali ukoliko se upravlja sa više od 10 vatrozida, to itekako predstavlja problem, odnosno vremenski je zahtjevno. Opisanim SDN pristupom štedi se na vremenu, odnosno na brzini rješavanja problema i odziva na neku dojavu koju je „okinuo“ događaj u mreži.

U slučaju mjerenje propusnosti između dva terminalna uređaja, koja se nalaze u tradicionalnoj mreži, potrebno je napraviti sljedeće korake:

1. Spojiti se na svaki terminalni uređaj posebno,
2. Instalirati alat koji generira promet te mjeri propusnost na spomenute uređaje,
3. Iskoristiti alat te očitati mjerenja.

Ukoliko treba napraviti mjerenje u SDN mreži kao što je mininet, potrebno je napisati jednu naredbu koja će biti odaslana prema kontroleru. Slika 17. predstavlja naredbu kojom se mjeri propusnost između H1 i H2.



Slika 17. Provjera propusnosti

Ukoliko se promijene vrijednosti kašnjenja i propusnosti na *linku*, na Slici 18. se može vidjeti kako se kroz softverski definiranu mrežu mjenjaju parametri na *linkovima*.

```
mininet@mininet-vm:~$ sudo mn --link tc,bw=10,delay=10ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit 10ms delay) (10.00Mbit 10ms delay) (h1, s1) (10.00Mbit 10ms delay) (10.00Mbit 10ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...(10.00Mbit 10ms delay) (10.00Mbit 10ms delay)
```

Slika 18. Provjera kašnjenja

Slika 19. prikazuje vrijednost RTT[12]-a (engl. *Round Trip Time*) otprilike 40ms što odgovara kašnjenju od 10ms kroz dva linka na putu u oba smjera (4 *10ms).

```
mininet> h1 ping -c10 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=50.2 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=48.1 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=50.4 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=48.9 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=47.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=45.9 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=45.7 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=43.1 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=43.3 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=43.1 ms

--- 10.0.0.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9021ms
rtt min/avg/max/mdev = 43.103/46.706/50.467/2.726 ms
```

Slika 19. Provjera kašnjenja s promjenjenjim vrijednostima

U ovakvom okruženju moguće je vrlo efikasno manipulirati osnovnim QoS parametrima pa tako Slika 20. prikazuje definiranje *jittera,* odnosno kolebanja kašnjenja.

[12] RTT: engl. *Round Trip Time*, ključan čimbenik za mjerenje vremena kašnjenja i učitavanja web stranica.

```
mininet@mininet-vm:~$ sudo mn --link tc,bw=10,delay=10ms,jitter=100ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit 10ms delay 100ms jitter) (10.00Mbit 10ms delay 100ms jitter) (h1, s1) (10.00Mbit 10ms delay 100ms j
itter) (10.00Mbit 10ms delay 100ms jitter) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...(10.00Mbit 10ms delay 100ms jitter) (10.00Mbit 10ms delay 100ms jitter)
*** Starting CLI:
```

Slika 20. Provjera jitter-a

Na Slici 21. vidljiv je utjecaj promjene *jitter*-a na RTT, a ovakvim prikazom i načinom rada može se smanjiti vrijeme potrebno za mjerenje QoS parametara u odnosnu na tradicionalan pristup. Također, moguće je vidjeti stvaran utjecaj parametara na sustav, što je u tradicionalnim mrežama mnogo teže postići.

```
mininet> h1 ping -c10 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=235 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=86.6 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=137 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=112 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=319 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=196 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=74.4 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=165 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=211 ms

--- 10.0.0.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9010ms
rtt min/avg/max/mdev = 0.069/154.149/319.675/87.131 ms
```

Slika 21. Provjera jitter-a s promjenjenim vrijednostima

Iz ovih jednostavnih naredbi vidi se velika prednost SDN mreže u odnosu na tradicionalne mreže. Danas je problem napraviti pravi test propusnosti u tradicionalnoj mreži jer spomenuta mreža nema softversku definiciju te naknadna nadogradnja takvu mrežu čini još kompleksnijom, što nikako nije slučaj SDN mreže, koja je po svojoj prirodi nadogradiva.

7. Analiza performansi SDN mreža

Kao što je ranije pojava VoIP[13] (engl. *Voice over Internet Protocol)* tehnologije uzrokovala velike promjene na telekomunikacijskom tržištu, danas su to softverski definirane širokopojasne mreže (engl. *Software Defined WAN*, SD-WAN) koje tržište osvajaju prvenstveno zbog uštede troškova u odnosu na tradicionalne mreže. Mijenjaju se tehnološke potrebe prema percepciji raznih telekomunikacijskih tvrtki i operatora. Softverski definirane širokopojasne mreže su specifične po primjeni umrežavanja softverski definiranih tehnologija koje se koriste na WAN[14] (engl. *Wide Area Network*) vezama za povezivanje poslovnih mreža velikih zemljopisnih udaljenosti, npr. za podružnice i podatkovne centre. Očekuje se znatan porast vrijednosti tržišta softverskih mreža u Europskoj Uniji do 2020. godine, a prema istraživanjima bi 50% vodećih tvrtki trebalo usvojiti takve vrste mreže u naredne tri godine. Pojavom upravljačke vrijednosti softverskog umrežavanja na postojećem tržištu širokopojasnih mreža, ekonomska ušteda više nije glavni pokretač tehnološke ekspanzije jer tvrtke mogu smanjiti složenost svoje mreže i dobiti mjerljivije podatke o prednostima kontrole, troškovima i funkcijivakog pojedinog elementa. Mrežni inženjeri poboljšavaju poslovne rezultate pomoću softverskih mreža, koristeći sposobnost centralnog upravljanja mrežom u realnom vremenu, što omogućuje izravno praćenje izmjena na mreži, segmentacije mreže, analitike i pojasne širine. *Big data* je često spominjani termin na telekomunikacijskom tržištu, međutim, sam kapacitet mreže nije jedini parametar koji je potrebno zadovoljiti kada se govori o kvaliteti mreže, puno je važnija sposobnost tvrtke da reagira i suprotstavi se potrebama tržišta. To je zapravo najveća performansa softverskih mreža koja u praksi nadilazi činjenicu o uštedi troškova. [33]

Analiza performansi u kontekstu računalnih mreža predstavlja obradu prikupljenih podataka s ciljem procjene odstupanja postojećih od željenih rezultata prilikom neke vrste usporedbe. Mjerenje performansi u postupku analize performansi ima ključnu

[13] VOIP: engl. *Voice over Internet Protocol* je komunikacijska tehnologija koja omogućava prijenos zvuka putem internet mreže.
[14] WAN: engl. *Wide Area Network* je kratica za engleski pojam koji se obično prevodi kao globalna mreža, a označava podatkovnu mrežu koja pokriva veće zemljopisno područje.

ulogu. Spomenut će se neka od najvažnijih mjerila performansi, koja ujedno predstavljaju i osnovne QoS[15] parametre:

- **Propusnost** izražava efektivnu brzinu prijenosa podataka izraženu brojem prenesenih bita u sekundi. Svaka aplikacija zahtjeva drugačiju razinu propusnosti, a nedovoljna razina propusnosti rezultirat će kašnjenjem u prijenosu.
- **Gubitak paketa** nastaje u slučaju kada je spremnik nekog od *router*-a u mreži pun, prouzročeno čekanjem paketa u redovima za usmjeravanje. U nekim slučajevima, ako je mreža na taj način konfigurirana, paketi koji kasne, biti će ispušteni. Ovaj parametar stvara probleme kod aplikacija koje se prenose u realnom vremenu.
- **Kašnjenje** označava vrijeme potrebno paketu sa stigne s kraja na kraj mreže. Osnovna mjerna jedinica je sekunda, ali premda sekunda označava preveliko kašnjenje u praksi se najčešće koristi ms – mili sekunda.

  Čimbenici koji utječu na veličinu kašnjenja su:

  - kašnjenje zbog kodiranja i dekodiranja
  - kašnjenje zbog komprimiranja i dekomprimiranja
  - kašnjenje zbog paketizacije i depaketizacije
  - kašnjenje zbog prijenosa na linku
  - kašnjenje zbog propagacije
  - kašnjenje zbog usmjeravanja u čvorovima
  - kašnjenje zbog čekanja u međuspremnicima rutera.
- ***Jitter* ili kolebanje kašnjenja**, definira se kao razlika u kašnjenju između susjednih paketa iste sesije. Vrijednosti *jitter*-a ovisit će o frekvenciji kojom se paketi šalju i fokusiraju se isključivo na kratkoročne događaje. [34]

7.1. Područja primjene SDN mreža

Kao što je ranije spomenuto, softverski definirano umrežavanje nudi nekoliko prednosti za tvrtke koje se pokušavaju preseliti u virtualno okruženje. Postoji mnoštvo slučajeva uporabe za različite organizacije, uključujući davatelje mobilnih usluga i usluga računalstva u oblaku, podatkovne centre, kao i velike razvojne kampuse. Za davatelje mobilnih usluga, softverski definirano umrežavanje nudi *bandwith* na zahtjev, što omogućava kontrolu veza mobilnih operatera i traženje dodatne širine

[15] QoS: engl. *Quality of Service* je kvaliteta usluge prikazana referentnim parametrima.

pojasa kada je to potrebno, kao i optimizaciju WAN-a. Za davatelje usluga računalstva u oblaku i podatkovne centre, virtualizacija mreža za višestruke stanare važan je slučaj uporabe jer nudi bolju upotrebu resursa i brže vrijeme obrade za izradu segregirane mreže. Razvojni kampusi obuhvaćaju kontrolu pristupa mreži i nadgledanje mreže prilikom korištenja programskih odredbi softverski definiranog umrežavanja. Jasni, dobro formirane slučajevi uporabe uvijek su imali važnu ulogu u izvršavanju usredotočenih inicijativa za razvoj softvera i uspješnih prodajnih programa. Razvojem SDx[16] (engl. *Software Defined Everything*) tehnologija, primjenjivi slučajevi uporabe postaju sve važniji i za dobavljače i za krajnje korisnike. SDx infrastruktura (kratica SDxI) odgovorna je za potporu i povezivanje korisnika i uređaja s uslugama, što znači da su problemi s kojima se susrećemo u mreži, eksponencijalno veći od problema umrežavanja od prije desetak godina. Nova internetska infrastruktura zahtijeva različitu razinu hardvera i softvera namijenjenih za pružanje usluge, automatizacije i fleksibilnosti koje zahtijevaju SDx aplikacije i računalstvo u oblaku. Dakle, sasvim različit skup primjena kako bi SDx tehnologija bila relevantna za telekomunikacijsko tržište danas. Slučajevi koherentnog korištenja ključni za usvajanje softverski definirane mreže i virtualizacije mrežnih funkcija (engl. *Network Functions Virtualization*, NFV[17]), važan su aspekt SDx infrastrukture. Ipak, definiranje slučajeva uporabe nije jednostavno jer razni proizvođači opreme, koji imaju udio u SDx platformi, definiraju tehnologije u svoje svrhe. Uspostavljanje široko razumljivih slučajeva uporabe SDxI pomoći će telekomunikacijskoj industriji postići produktivnije razgovore o potrebama korisnika, tehničkim zahtjevima i relevantnim poslovnim rješenjima za SDN i NFV.

Kada se govori o novijim područjima primjene SDxI, ne misli se pritom da su tradicionalni načini umrežavanja zanemarivi ili neupotrebljivi. Primjerice, neki mrežni uređaji proizvedeni prije više od 50 godina, još se uvijek upotrebljavaju jer jednako učinkovito rješavaju iste probleme koji su im prijetili i prije 50 godina. Dakle, tradicionalni način umrežavanja nije postao ništa manje važan pojavom SDN-a, upotrebljavat će se još dugi niz godina i trebat će vremena kako bi koncept softverskog umrežavanja u potpunosti zaživio. Uspon SDx-a otvorio je brojna

[16] SDx: engl. *Software Defined Everything* je bilo koja fizička stavka ili funkcija koja se može izvesti ili automatizirati pomoću softvera.

[17] NFV: engl. *Network Functions Virtualization* je novi način dizajniranja, implementacije i upravljanja mrežnim uslugama. Odvaja mrežne funkcije, poput mrežnog adresiranja, vatrozida, otkrivanja ranjivosti, DNS servera itd. od fizičkih hardverskih mrežnih uređaja.

nepoznata infrastrukturna pitanja. Nastala SDx infrastruktura u osnovi mijenja organizacije i uzrokuje pomak kod kritičnih uloga funkcionalnih rukovoditelja, tehničkih arhitekata i razvojnih programera. Konkretni slučajevi uporabe ljudima na tim ulogama pružaju potrebne informacije za razumijevanje SDx tehnologija i donošenje odluka o kupnji. Kod tradicionalnih mrežnih modela, uporaba konkretnih slučajeva bila je jasnija jer bi korisnik odabrao mali dio arhitekture kako bi riješio svoje poslovne probleme. U SDx svijetu aplikacije su toliko prilagođene da su postale konkurentni razlikovni faktori. Na primjeru danas vrlo poznatih svjetskih usluga vidljivo je na koji način funkcionira SDx. Stranica za najam smještaja, AirBnB, je izgradila aplikacije usmjeravanja pozadinskog ureda i aplikacije usmjeravanja pozivnih centara koji čine okosnicu svog poslovanja, dok je mreža prijevoznika, Uber, izgradila sustave komunikacije i raspoređivanja koji služe kao središte živčanog sustava vlastitog poslovanja. Korisnici poput ovih u budućnosti bit će daleko manje zainteresirani za značajke u definiranju jasnih slučajeva koji će im pomoći u rješavanju najhitnih izazova i vrednovati potencijalne tehnologije. [35]

7.1.1. NFV koncept

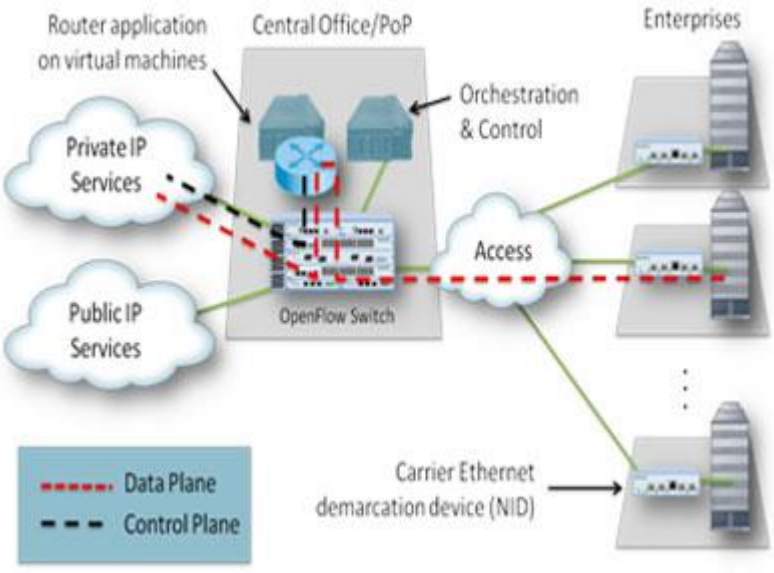
Definiranje slučajeva uporabe nije jednostavan proces. Ne samo da svi sudionici telekomunikacijskog tržišta koriste blago drugačiju terminologiju kako bi opisali neznatno različite potrebe i potencijalne načine kako bi se te potrebe ispunile, nego je također potrebno kategorizirati određene slučajeve uporabe. Iz brojnih istraživanja proizlazi šest najčešćih zajedničkih slučajeva uporabe SDN-a i NFV-a, koji najbolje funkcioniraju u kombinaciji:

1. Kontrola pristupa mreži: Postavljanje odgovarajućih povlastica za korisnike ili uređaje koji pristupaju mrežama, uključujući ograničenja kontrole pristupa, uključivanje lanaca usluga i odgovarajuću kvalitetu usluge. Da bi se osigurala sigurnost i usklađenost, a istodobno postigla visoka iskoristivost, potrebno je pratiti korisnike i uređaje kada se povezuju s različitih dijelova mreže.
2. Mrežna virtualizacija: Stvaranje virtualne mreže na fizičkoj mreži, omogućujući velikom broju mreža pokretanje preko fizičke mreže. Ona može obuhvaćati više rekova[18] u podatkovnom centru ili više udaljenih lokacija ukoliko je to

[18] Rek: engl. *rack* u telekomunikacijskoj terminologiji označava metalni ormar za smeštaj opreme u podatkovnom centru ili serverskoj sobi.

potrebno. Uključuje mrežno upravljanje i izolacije, kao i implementaciju ubrzanja ili sigurnosnih usluga.

3. Virtualna granica korisnika: Virtualiziranje granice korisnika moguće je ili stvaranjem virtualne platforme na prostoru koje zauzima korisnik ili povlačenjem funkcija bliže jezgri mreže na virtualnoj multinacionalnoj platformi. Platforma može biti postavljena u regionalnom ili središnjem podatkovnom centru kao i kod krajnjeg korisnika
4. Dinamičan međuodnos: Izrađuju se dinamične veze između lokacija, uključujući veze između podatkovnih centara, poduzeća i drugih lokacija tvrtke. Na vezama se primjenjuje odgovarajući QoS i pojasna širina.
5. Virtualne jezgre / agregacijske mreže: Virtualiziraju se osnovni sustavi za davatelje usluga, uključujući mobilnu infrastrukturu za podršku kao što su vIMS, vEPC, kao i NFV GiLAN infrastrukturu.
6. Optimizacija podatkovnog centra: Optimizacija mreže pomoću SDN-a i NFV-a, u svrhu otkrivanja performansi aplikacija i uzimajući u obzir afinitete i skokovite promjene opterećenja konfiguracije mreže.



Slika 22. NFV koncept

Izvor [36]

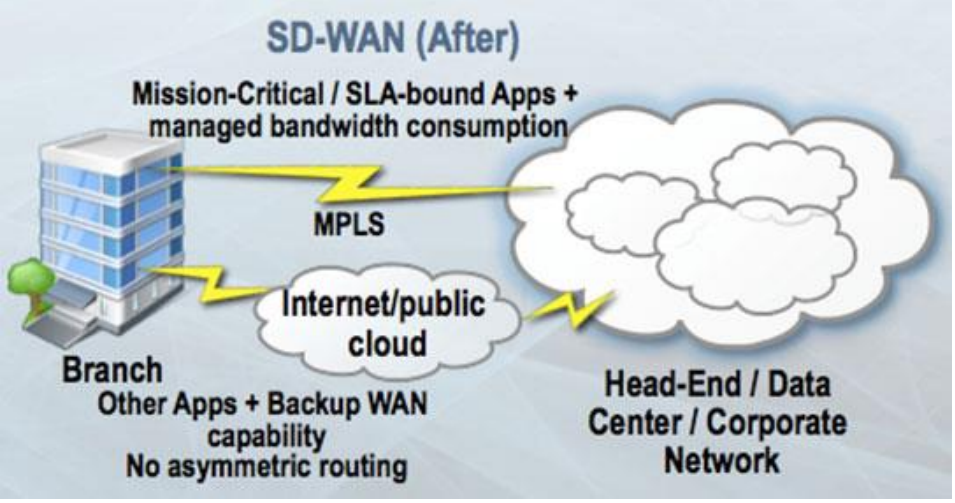
U ovom dijelu važno je dotaknuti se i performansi spomenutog koncepta virtualizacije mrežnih funkcija. NFV (Slika 22.) virtualizira mrežne usluge putem softvera kako bi operaterima omogućio sljedeće:

- Smanjenje CapEx-a[19]: Smanjenje potrebe za kupnjom hardverskih uređaja u svrhu povećanja plaća ljudskim resursima kako bi se eliminirali prekomjerni troškovi izgradnje mreže.
- Smanjenje OpEx-a[20]: Smanjenje potrebe za prostorom, snagom i hlađenjem opreme te pojednostavljenje izvedbe i upravljanja mrežnim uslugama kako bi se eliminirali prekomjerni troškovi održavanja.
- Ubrzanje plasiranja na tržište: Smanjenje vremena potrebnog za implementaciju novih usluga kako bi se podržali promjenjivi poslovni zahtjevi, iskoristile nove tržišne prilike i povećao povrat ulaganja novih usluga. Također smanjenje rizika povezanih s osmišljanjem i izvođenjem novih usluga, kako bi davatelji usluga jednostavno odredili što najbolje odgovara potrebama korisnika.
- Agilnost i fleksibilnost: Omogućavanje povećanja ili smanjenja razine usluge za rješavanje promjenjivih zahtjeva korisnika. Također i podržavanje inovacija i omogućavanje isporuke usluge putem softvera na bilo kojem industrijski standardnom hardverskom uređaju poslužitelja. [37]

7.1.2. SD-WAN

Softverski definirana širokopojasna mreža (kratica SD-WAN) specifičan je način primjene softverski definiranog umrežavanja koja se primjenjuje na WAN mreže, koje se koriste za povezivanje mreža na velikim geografskim udaljenostima, uključujući udaljene podružnice i podatkovne centre. WAN se, primjerice, može koristiti za spajanje više poslovnica na središnju mrežu tvrtke ili za povezivanje podatkovnih centara udaljenih lokacija. U prošlosti su WAN veze koristile tehnologiju koja zahtijeva zasebnu hardversku infrastrukturu i uređaje, dok se pojavom SD-WAN-a nadzor mreže nastoji premjestiti u "oblak" uz pomoć softverskog pristupa (Slika 23.).

[19] CapEx: engl. *Capital expenditure*, kapitalni troškovi ili troškovi izgradnje.
[20] OpEx: engl. *Operating expenditure*, operativni troškovi ili troškovi održavanja.



Slika 23. SD-WAN koncept

Izvor [38]

Poslovni korisnici zahtijevaju fleksibilnije, otvorene i WAN tehnologije temeljene na računalstvu u oblaku, umjesto instaliranja specijalizirane WAN tehnologije koja često uključuje skupe, fiksne sklopove ili posjedovanje hardverske opreme. Mnoge novije SD-WAN ponude, kao što su T-1 ili MPLS, mogu se koristiti za poboljšanje i osiguravanje internetske povezanosti, što ih čini konkurentnijim i skupljim WAN tehnologijama. U nekim slučajevima SD-WAN koristi internetske širokopojasne veze za zamjenu skupih rješenja. Tehnologija virtualizacije može primijeniti tehnologiju sigurnosti i virtualne privatne mreže na širokopojasne internetske veze, što ih čini sigurnijima. SD-WAN također ima mogućnost uklanjanja potencijalno skupog hardvera za usmjeravanje prometa, pružanjem povezivanja i usluga putem računalstva u oblaku. Nove tehnologije SD-WAN također su fleksibilnije, budući da se SD-WAN povezivost može kontrolirati putem računalstva u oblaku, korisnik može prilagoditi ili povećati povezanost tijekom vremena veće potražnje. Glavni cilj tehnologije SD-WAN je pružanje sigurne i jednostavne WAN veze omogućene za računalstvo u oblaku i otvorene za softverske tehnologije. To se može koristiti za isporuku privatne WAN usluge ili za poslovne usluge kao što su VPN, optimizacija WAN-a i kontrola isporuke aplikacija.

Brojni vodeći proizvođači opreme su promijenili pristup tržištu nakon uočavanja performansi SD-WAN-a. Na primjer, tvrtke kao što su Cisco i Riverbed, koje proizvode specijalizirane uređaje za WAN povezivanje, sada se više usredotočuju na ponudu WAN mreža temeljenih na računalstvu u oblaku kao odgovor na taj novi trend. Očekuje se rast interesa tijekom sljedećih nekoliko godina. Ono što je započelo kao rješenje za globalnu povezanost podružnica i podatkovnog centra uz manje hardverske opreme, proširuje se na širok spektar ponuda i tehnologija SD-WAN-a, uključujući VPN, sigurnost, optimizaciju WAN-a, NaaS i kontrolu pravila privatnosti. [39]

7.2. Poslovne prednosti SDN mreža

SDN nudi centraliziranu, programibilnu mrežu koja može pružiti rješenja na promjenjive potrebe poduzeća. Također, vidljive su sljedeće tehničke i poslovne performanse:

- Izravna programibilnost: SDN mrežna pravila izravno se mogu programirati, jer su funkcije upravljanja mrežom odvojene od funkcija prosljeđivanja, što omogućuje da se mreža programski konfigurira pomoću vlasničkih ili open source alata za automatizaciju, uključujući OpenStack, Puppet i Chef.
- Centralizirano upravljanje: Mrežna inteligencija logički je centralizirana u softveru SDN kontrolera koji održava globalni pogled na mrežu.
- Smanjenje CapEx-a: SDN potencijalno ograničava potrebu za kupnjom hardverske opreme za izgradnju mreže hardvera koji se temelji na ASIC-u, a umjesto toga podržava tzv. *pay-as-you-grow* modele.
- Smanjenje OpEx-a: SDN omogućuje algoritamski nadzor mrežnih elemenata (kao što su hardverski ili softverski *router*-i i *switch*-evi) koji su opće programabilni, što olakšava projektiranje, implementaciju, upravljanje i mjerenje referentnih parametara mreža. Sposobnost automatizacije opskrbe optimizira dostupnost i pouzdanost usluge smanjenjem ukupnog vremena upravljanja i prilike za ljudsku pogrešku.
- Pružanje agilnosti i fleksibilnosti: SDN omogućuje organizacijama brzu implementaciju novih aplikacija, usluga i infrastrukture kako bi udovoljile promjenjivim poslovnim ciljevima.

- Omogućavanje inovacija: SDN omogućuje organizacijama stvaranje novih vrsta aplikacija, usluga i modela poslovanja koji mogu ponuditi nove prihode i veću vrijednost mreže. [40]

Tablica 3. Komparativna analiza tradicionalne i SDN mreže

	Tradicionalna mreža	**SDN mreža**
Dinamika	Postaje vrlo složena kada se radi o više uređaja i mobilnom okruženju.	Brzo se prilagođava promjenama poslovnih potreba.
Sigurnosna politika	Kod aplikacija za širokopojasnu mrežu potrebne su izmjene sigurnosnih politika.	Sigurnosne politike postaju pojednostavljene i dosljedne.
Skalabilnost	Skalabilnost mreže postaje neodrživa s povećanjem mrežnih uređaja.	Jednostavno skalabilna zbog dosljednog centraliziranog upravljanja.
Kontrola uređaja	Nepotpuna kontrola uređaja, jer proizvođači opreme predlažu promjene.	Proizvođači opreme nemaju utjecaja, kontrolu uređaja u potpunosti obavlja korisnik.
Metoda pristupa	Mrežni protokol SNMP[21]	*OpenFlow* protokol
Korištenje informacija	Potrebna zasebna upravljačka jedinica na svakom mrežnom uređaju.	Upravljačka i podatkovna razina su odvojene i imaju uspostavljenu međusobnu komunikaciju.

Izvor [41]

[21] SNMP: engl. *Simple Network Management Protocol*, jednostavan mrežni protokol za nadzor i upravljanje.

Za kraj, Tablicom 3. prikazana je komparativna analiza tradicionalne i SDN mreže, kako bi se još jednom istaknule neke od glavnih prednosti implementacije SDN rješenja.

8. Zaključak

Softverski definirane mreže predstavljaju oblik mrežne arhitekture koji nastoji prilagoditi mrežu današnjim potrebama tržišta, odvajanjem upravljačkog dijela mreže od dijela prosljeđivanja, stvarajući programabilnu infrastrukturu koja se razlikuje od fizičkih uređaja i dosadašnjih tradicionalnih mreža. Sve veća je odgovornost na brzini obrade podataka i samom upravljanju paketima u mrežnim uređajima. Upravo ideja razdvajanja mrežne infrastrukture na zasebne dijelove, prikazuje srž koncepta softverskog upravljanja mrežom. U takvom sustavu, glavnu ulogu ima središnja upravljačka jedinica, a uz nju programska sučelja, *Southbound* API i *Northbound* API. SDN kontroleri nude centralizirani prikaz cjelokupne mreže i omogućuju mrežnim administratorima upravljanje osnovnim sustavima. *Southbound* API-ji koriste se za prijenos informacija na *router*-e i *switch*-eve. *OpenFlow* se smatra prvim standardom u SDN-u, a izvorno je bio *Southbound* API i ostaje jedan od najčešće korištenih protokola. *Northbound* API-ji koriste se za komunikaciju s aplikacijama i pomažu mrežnim administratorima da programski oblikuju promet i implementiraju usluge. Kada se govori o planiranju mreže, ono zahtijeva primjenu znanstvenih metoda, a time i izvođenje planiranja treba izvoditi na visokoj stručnoj razini, uz najužu koordinaciju tehnike i tehnologije te gospodarstva i financijskih institucija. Također i samo ignoriranje SDN-a predstavlja rizik za telekomunikacijske organizacije, kao i za stručnjake u području telekomunikacija. Rizik za telekomunikacijske organizacije je u tome što neće biti u stanju riješiti probleme za koje je SDN u prvom redu dizajniran, čega je posljedica nedostatak konkurentnosti na tržištu. Rizik za stručnjake u području telekomunikacija je nedostatak edukacije vezane za ovaj način umrežavanja, a samim tim neće biti konkurentni na tržištu rada. Zbog toga se mora razviti dobar plan za implementaciju SDN-a na telekomunikacijskom tržištu. S obzirom na brze promjene u tehnološkom razvoju, pa tako i razvoju SDN-a, bilo koji plan implementacije razvijat će se dalje tijekom vremena. Korištenjem programskog alata GNS3 prikazana je konfiguracija i simulacija SDN mreže i neki od mogućih scenarija. Primjerice, ukoliko se promijene vrijednosti kašnjenja i propusnosti na *linku*, moguće je vidjeti kako se kroz softverski definiranu mrežu mjenjaju parametri na *linkovima*. U takvom programskom okruženju moguće je vrlo efikasno manipulirati osnovnim QoS parametrima, a smanjuje se i vrijeme potrebno za njihovo mjerenje u

odnosnu na tradicionalan pristup. Također, moguće je vidjeti stvaran utjecaj parametara na sustav, što je u tradicionalnim mrežama mnogo teže postići.

Softverski definirano umrežavanje nalazi brojna područja primjene i tržište polako uviđa prednosti korištenja i ulaganja u ovakav način umrežavanja. Uzme li se u obzir dugogodišnje korištenje mrežnih arhitektura koje bilježe brojne propuste, dakako je potrebna edukacija i analiza što više performansi ovakvog načina umrežavanja kako bi dobio naklonost tržišta. Razlog korištenja zastarjelih metoda leži jednim dijelom u prihodima koje donose mrežnim operatorima, a drugim dijelom u vodstvu tvrtki i organizacija koje nije spremno uložiti u inovacije jer se vode politikom zašto nešto mijenjati dok obavlja svoju funkciju.

Literatura

[1] Ranjan, P.: A Survey of Past Present and Future of Software Defined Networking, Hal, Tokyo, Japan, 2014.

[2]URL:https://www.google.hr/search?q=LEGACY+VS+SDN+INFRASTRUCTURE&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiB19n_rZXcAhVCDSwKHa3jD-4Q_AUICigB&biw=1366&bih=645#imgrc=TdX9vl0qDZ_h3M: (pristupljeno: svibanj 2018.)

[3] URL: https://www.sdxcentral.com/sdn/definitions/what-the-definition-of-software-defined-networking-sdn/ (pristupljeno: svibanj 2018.)

[4] Nadeau, T.D.; Gray, K.: SDN: Software Defined Networks, First Edition, O'Reilly Media, Inc., USA, 2013.

[5]URL:https://www.google.hr/search?q=sdn+controller+platform&source=lnms&tbm=isch&sa=X&ved=0ahUKEwinx5ydsJXcAhUH6CwKHVpsB8oQ_AUICigB&biw=1366&bih=645#imgrc=B6P8dCayv3BjFM: (pristupljeno: lipanj 2018.)

[6] Braun, W.; Menth, M.: Software-Defined Networking Using OpenFlow: Protocols Applications and Architectural Design Choices, Future Internet, Tuebingen, Germany, 2014.

[7] Marschke, D.; Doyle, J.; Moyer, P.: SDN: Anatomy of Openflow, Volume I, Lulu Press, Inc., Morrisville, USA, 2015.

[8] URL: https://www.juniper.net/us/en/solutions/sdn/what-is-sdn/ (pristupljeno: lipanj 2018.)

[9] URL: https://docs.microsoft.com/en-us/windows-server/networking/sdn/software-defined-networking (pristupljeno: lipanj 2018.)

[10]URL: https://www.cisco.com/c/en/us/solutions/software-defined-networking/overview.html (pristupljeno: lipanj 2018.)

[11] URL: https://www.opennetworking.org/sdn-definition/ (pristupljeno: lipanj 2018.)

[12]URL: https://academy.gns3.com/p/sdn-and-openflow-introduction (pristupljeno: lipanj 2018.)

[13] URL: https://www.gns3.com/news/article/sdn-101-mininet-openflow-and-gns (pristupljeno: lipanj 2018.)

[14] Guizani, M.; Rayes, A.; Khan, B.; Al-Fuqaha, A.: Network Modeling and Simulation: A Practical Perspective, 1st Edition, Wiley, Idaho, USA, 2010

[15] Kocharians, N., Paluch, P.; Vinson, T.: CCIE Routing and Switching v5.0 Official Cert Guide Library, 5th Edition, Cisco Press, San Jose, USA, 2014.
[16] Nugroho, A.S.; Safitri, Y.D.; Setyawan, T.A.: Comparison Analysis of Software Defined Network and OSPF Protocol Using Virtual Media, Hal, Nagoya, Japan, 2017.
[17] Laponina, O.R.; Sizov, M.R.: Laboratory Bench for Testing the Integration Capabilities of SDN Networks and Traditional Networks, Lomonosov Moscow State University, Moscow, Russian Federation, 2017.
[18] Cassongo, A.B.: The Comparison of Network Simulators for SDN, Springer, Berlin, Germany, 2016.
[19]URL:https://www.google.hr/search?biw=1366&bih=645&tbm=isch&sa=1&ei=CBRFW8eQA4XWsAHFw7LgDw&q=SDN&oq=SDN&gs_l=img.3..35i39k1l2j0i19k1l8.114508.115091.0.115382.3.3.0.0.0.0.197.416.0j3.3.0....0...1c.1.64.img..0.3.414...0j0i67k1.0.PCo-FLPjMX0#imgrc=c1K9oXaXbTzUWM: (pristupljeno: svibanj 2018.)
[20] Göransson, P., Black, C.: "Software Defined Networks - A Comprehensive Approach", Elsevier, Inc., Waltham, USA, 2014.
[21] Open Network Foundation: "SDN Architecture Overview", White Paper, ONF, Menlo Park, USA, 2014.
[22]URL:https://www.google.hr/search?q=sdn+domain&source=lnms&tbm=isch&sa=X&ved=0ahUKEwikidXbrpXcAhVG_iwKHYaTAyUQ_AUICigB&biw=1366&bih=645#imgrc=0qM_yRoO_r8I2M: (pristupljeno: lipanj 2018.)
[23]URL: https://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-59/161-sdn.html (pristupljeno: svibanj 2018.)
[24]URL: http://flowgrammable.org/sdn/openflow/ (pristupljeno: lipanj 2018.)
[25]URL:https://www.google.hr/search?q=openflow+ssl&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiZjK_msJXcAhUjiKYKHbZqCK4Q_AUICigB&biw=1366&bih=645#imgrc=5d6jgLX6N1zF-M: (pristupljeno: svibanj 2018.)
[26]URL:https://www.google.hr/search?biw=1366&bih=645&tbm=isch&sa=1&ei=XBNFW9_DEoScsAG07pmgCA&q=flow+table+pipeline&oq=FLOW+TABLE+&gs_l=img.1.0.35i39k1j0i19k1l9.2006.2006.0.4055.1.1.0.0.0.0.141.141.0j1.1.0....0...1c.1.64.img..0.1.141....0.bcWPSOKgKYY#imgrc=wExsOEJkcfae5M: (pristupljeno: lipanj 2018.)
[27]URL:https://www.google.hr/search?q=openflow&source=lnms&tbm=isch&sa=X&ved=0ahUKEwi9vKPmrJXcAhWIOSwKHZfLCasQ_AUICigB&biw=1366&bih=645#imgrc=MOdWu5FR-TcPOM: (pristupljeno: lipanj 2018.)

[28]URL:https://www.google.hr/search?biw=1366&bih=645&tbm=isch&sa=1&ei=XBNFW9_DEoScsAG07pmgCA&q=flow+table+pipeline&oq=FLOW+TABLE+&gs_l=img.1.0.35i39k1j0i19k1l9.2006.2006.0.4055.1.1.0.0.0.0.141.141.0j1.1.0....0...1c.1.64.img..0.1.141....0.bcWPSOKgKYY#imgrc=TMxJ9aWEOYqYNM: (pristupljeno: lipanj 2018.)
[29] Open Network Foundation: "OpenFlow Switch Specification Version 1.5", ONF, Menlo Park, USA 2014.
[30]URL:http://e-student.fpz.hr/Predmeti/2002/Planiranje_TK_mreza/Materijali/Opcenito_o_planiranju.pdf (pristupljeno: svibanj 2018.)
[31] Metzler, J., Metzler, A.: "The 2013 Guide to Network Virtualization and SDN", Webtorials Analyst Division, Redhat, Boston, USA, 2013.
[32] Vissicchio, S., Vanbever, L., Bonaventure, O.: "Opportunities and Research Challenges of Hybrid Software Defined Networks" in Proceedings of the 13th ACM Workshop on Hot Topics in Networks, Iternational Journal, Zurich, Switzerland, 2014, pp. 70-75
[33]URL: http://www.skvid.net/voip-blog-cloud-connected/25-voip/287-novost-na-tr%C5%BEi%C5%A1tu-softverski-definirane-%C5%A1irokopojasne-mre%C5%BEe (pristupljeno: lipanj 2018.)
[34]URL:https://estudent.fpz.hr/Predmeti/T/Tehnologija_telekomunikacijskog_prometa_I/Materijali/10_predavanje.pdf (pristupljeno: kolovoz 2018.)
[35] URL: https://www.sdxcentral.com/cloud/definitions/software-defined-everything-part-5-sdx-use-cases/ (pristupljeno: svibanj 2018.)
[36]URL:https://www.google.hr/search?q=nfv&source=lnms&tbm=isch&sa=X&ved=0ahUKEwi3hL7cr5XcAhUBWSwKHYvnDOoQ_AUICigB&biw=1366&bih=645#imgrc=F9c3yU-ymYr3JM: (pristupljeno: lipanj 2018.)
[37]URL:https://www.sdxcentral.com/nfv/definitions/whats-network-functions-virtualization-nfv/ (pristupljeno: svibanj 2018.)
[38]URL:https://www.google.hr/search?q=sdn+wan&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj91cL9r5XcAhWKECwKHS0OAVMQ_AUICygC&biw=1366&bih=645#imgrc=D72cadu2llvfOM: (pristupljeno: lipanj 2018.)
[39] URL: https://www.sdxcentral.com/sd-wan/definitions/software-defined-sdn-wan/ (pristupljeno: svibanj 2018.)
[40] Kepes, B.: "SDN meets the real-world: Implementation benefits and challenges", GIGAOM RESEARCH, Giga Omni Media, Inc., San Francisco, USA, 2014.

[41]URL:https://www.google.hr/search?biw=1366&bih=641&tbm=isch&sa=1&ei=FKyBW5qHOuCrwShg67QCA&q=sdn+vs+traditional+network+performances&oq=sdn+vs+traditional+network+performances&gs_l=img.3...268684.273019.0.273230.13.13.0.0.0.0.236.1323.0j7j2.9.0....0...1c.1.64.img..4.2.256...0i30k1j0i8i30k1.0.hH037kcGuhA#imgrc=gF5DE7wIMHxOOM: (pristupljeno: kolovoz 2018.)
[42] Youtube video sadržaj: https://www.youtube.com/watch?v=DiChnu_PAzA (pristupljeno: srpanj 2018.)
[43] Youtube video sadržaj: https://www.youtube.com/watch?v=rYW7kQRyUvA (pristupljeno: srpanj 2018.)
[44] Youtube video sadržaj: https://www.youtube.com/watch?v=l25Ukkmk6Sk&t=1s (pristupljeno: srpanj 2018.)
[45] Youtube video sadržaj: https://www.youtube.com/watch?v=FyV4MoQ3T0I (pristupljeno: srpanj 2018.)
[46] Youtube video sadržaj: https://www.youtube.com/watch?v=TD5wmoD7XOE (pristupljeno: srpanj 2018.)
[47] Youtube video sadržaj: https://www.youtube.com/watch?v=jmlgXaocwiE (pristupljeno: srpanj 2018.)
[48] Youtube video sadržaj: https://www.youtube.com/watch?v=CPasnNg9Z4I (pristupljeno: srpanj 2018.)
[49] Youtube video sadržaj: https://www.youtube.com/watch?v=yHUNeyaQKWY (pristupljeno: srpanj 2018.)

Popis kratica i akronima

API - engl. *Application Programming Interface*

ARP - engl*. Address Resolution Protocol*

BoS - engl*. Bottom of Stack*

CapEx - engl. *Capital expenditure*

CLI - engl. *Command Line Interface*

GNS3 - engl. *Graphical Network Simulator-3*

ICMP - engl*. Internet Control Message Protocol*

LAN-a - engl. *Local Area Network*

MBI-a - engl. *Management Buy-In*

MPLS - engl*. Multiprotocol Label Switching*

NFV - engl. *Network Functions Virtualization*

ONF - engl*. Open Networking Foundation*

OpEx - engl. *Operating expenditure*

POC-a - engl. *Proof of Concept*

QoS - engl. *Quality of Service*

SCTP - engl*. Stream Control Transmission Protocol*

SDN - engl. *Software Defined Networking*

SDNi - engl. *Software Defined Networking interface*

SD-WAN - engl. *Software Defined WAN*

SDx - engl. *Software Defined Everything*

SNMP - engl. *Simple Network Management Protocol*

SPOF - engl. *Single Point of Failure*

SSL - engl. *Secure Sockets Layer*

TLS-a - engl. *Transport Layer Security*

TTL - engl*. Time To Live*

VoIP - engl. *Voice over Internet Protocol*

WAN - engl. *Wide Area Network*

Popis slika i tablica

Popis slika

Slika 1. Razlika između konvencionalne i SDN infrastrukture.....3
Slika 2. Logički prikaz SDN mreže.....4
Slika 3. Arhitektura SDN mreže.....7
Slika 4. Prikaz SDN domena.....10
Slika 5. Prikaz komponenti OpenFlow protokola.....12
Slika 6. Prikaz arhitekture OpenFlow protokola.....14
Slika 7. Prikaz OpenFlow switch-a.....14
Slika 8. Prikaz tablice usmjeravanja OpenFlow protokola.....15
Slika 9. Flow – table pipeline.....18
Slika 10. Izbornik emuliranih uređaja.....32
Slika 11. Radna površina GNS3 emulatora.....33
Slika 12. Terminal (CLI).....33
Slika 13. Fizički prikaz simuliranog sustava.....34
Slika 14. Logički prikaz simuliranog sustava.....35
Slika 15. Kreiranje mininet mreže.....36
Slika 16. Provjera gubitka paketa.....38
Slika 17. Provjera propusnosti.....39
Slika 18. Provjera kašnjenja.....40
Slika 19. Provjera kašnjenja s promjenjenjim vrijednostima.....40
Slika 20. Provjera jitter-a.....41
Slika 21. Provjera jitter-a s promjenjenim vrijednostima.....41
Slika 22. NFV koncept.....46
Slika 23. SD-WAN koncept.....48

Popis tablica

Tablica 1. Funkcija komponenti OpenFlow protokola.....13
Tablica 2. Poruke OpenFlow protokola.....21
Tablica 3. Komparativna analiza tradicionalne i SDN mreže.....50